The COMPLETE

PART 2

SPECTRUM

All you want to know about the world's best-selling computer

A Database Publication

Microdrives v discs: all the facts

Memory expansion made easy

Animation: how to get things moving

Learn to program in Basic and machine code

SAVE £££'s on Spectrum products —see centre pages

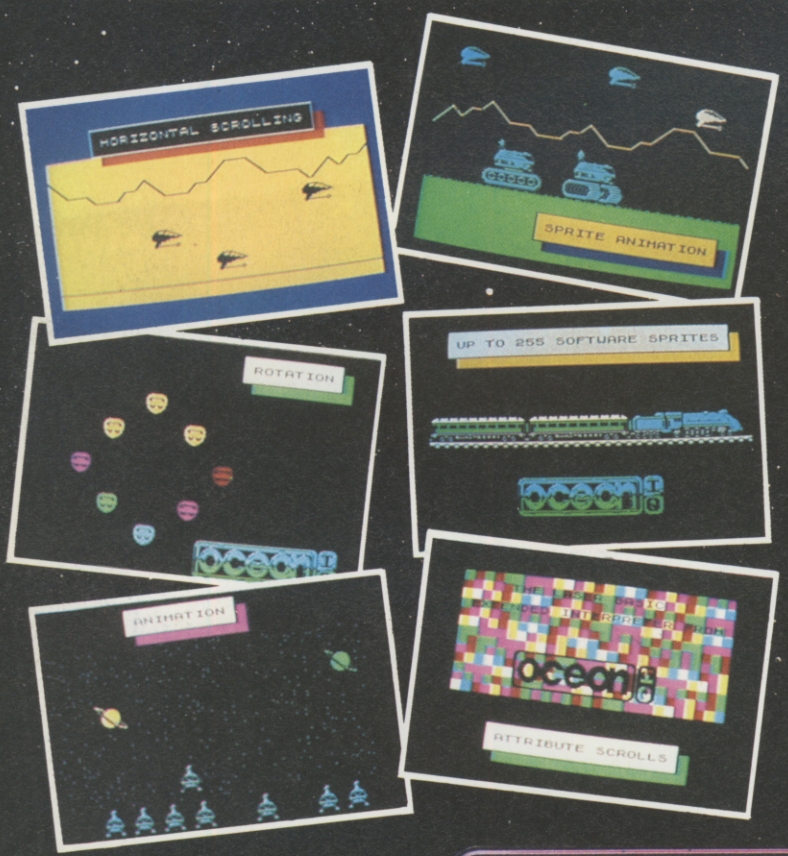
In six monthly parts

£1.50

Laser Basic

TURN YOUR COMPUTER INTO A PROFESSIONAL GAMES WRITING MACHINE FOR FUN AND PROFIT!

LASER BASIC adds 100 new commands to Sinclair Basic. These extended commands are semi-compiling so graphic animation is extremely fast. LASER BASIC includes, extended interpreter, sprite/graphic designer, fully documented program and 2 sets of pre-defined sprites.



- Up to 255 software sprites can be defined, each with its own user selectable dimensions (up to 7 screens wide!)
- Operations can be carried out, on or between screen windows, sprites and sprite windows.
- Sprites can be block 'PUT' or can use one of three logical operations – AND, OR and XOR.
- Sprites and screen windows can be pixel scrolled in any direction, mirrored, enlarged, spun, inverted or cleared.
- Procedures with local variables and parameter passing.
- TRON and TROF (trace facility).
- 16 bit PEEK and POKE.
- RENUMBER and REM renumber.
- Non destructive MOVE with 2 frame animation.
- Collision detection and pattern recognition facilities.

OUT NOW FOR THE SPECTRUM 48K/ SPECTRUM+ ON CASSETTE

£14.95
Microdrive compatible.

ALSO AVAILABLE NOW!

Laser Compiler
• Using the Laser Compiler your program runs even faster and can be produced to stand alone.
• Programs can be marketed with no royalities to pay (once you have bought a compiler your work is your own.)
ocean iQ

Spectrum 48K/ Spectrum+ **£9.95** on cassette. Microdrive compatible.

A new age dawns! The arrival of LASER BASIC: first in a powerful range of development tools for fast programming, brought to you with the combined skill and resources of Ocean and Oasis – (producers of "White Lightning".)
Other, easy to use products in this expanding range will include screen artist/designers, music composers and machine code emulators to make your programming more rewarding and exciting.

ocean iQ INTERACTIVE SOFTWARE

Ocean iQ Software is available from selected branches of:
Boots, WHSMITH, John Menzies, WOOLWORTH, LASKYS, Rumbelows, Greens,
Spectrum Shops and all good software dealers. Trade enquiries welcome.

6 Central Street, Manchester M2 5NS. Telephone: 061-832 6633. Telex: 669977 Oceans G.

In Part Two...

38 Beginners
Variables come under the microscope. We show how to use them in your first program.

42 Discovering Discs
Planning on buying a disc drive? Here's all you need to make up your mind.

45 Microdrives
The facts about them and how to make the most of the additional commands.

52 Monitoring the video
Confused by composite video? Helpful hints to bring better colour to your screen.

54 INK and PAPER
All the colours of the rainbow. The Spectrum's shades in graphic detail.

57 Ready reference
Get the facts at your fingertips as we summarise the Spectrum's graphics commands.

58 Take a byte
Take the mystery out of the way your Spectrum handles binary numbers.

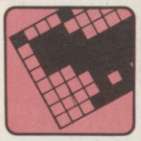
61 Analysis
A straightforward graphics program simply explained in line by line detail.

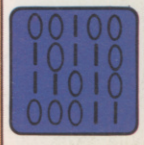
62 Get things moving
Animation can enliven any screen display – and it's much easier than it looks.

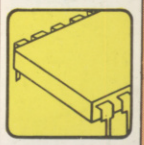
66 Machine code
Start machine code programming by learning how to PEEK, POKE and CALL.

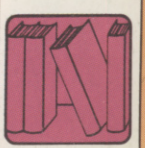
70 Ram Upgrade
Short of memory? Here are some of the options to expand your RAM power.

72 Looking into books
Read all about it. A personal selection of the best Spectrum books now available.









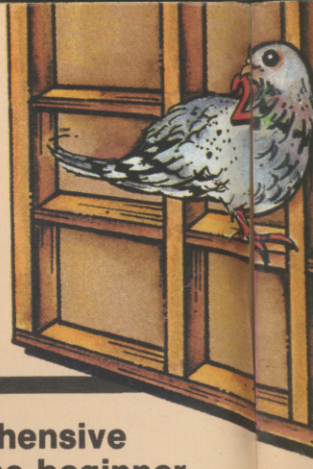
The COMPLETE SPECTRUM
All you want to know about the world's best-selling computer

is published by
Database Publications Ltd,
Europa House, 68 Chester Road,
Hazel Grove, Stockport SK7 5NY.
Tel: 061-456 8383

© Database Publications Ltd. No material may be reproduced in whole or in part without written permission. While every care is taken, the publishers cannot be held responsible for any errors in articles, listings or advertisements.

News trade distribution by:
Europress Sales & Distribution Ltd,
11 Brighton Road, Crawley,
West Sussex RH10 6AF.
Tel: 0293 27053.

Venture forth with variables



Part II of our comprehensive hands-on guide for the beginner

GETTING the Spectrum to do things is easy, so long as you know the words to use and how to use them properly. By now:

```
PRINT "a string"
PRINT 2*3
PRINT "My name is Peter"
PRINT (2+5)*7–1
PRINT 2+5*(7–1)
```

should give you no problems. If, however, you find it difficult to explain why the last two sums give different answers despite being made up of the same numbers, read the panel on operator precedence below.

LETting on about variables

On the L key of your Spectrum you'll see the keyword LET. When L is pressed in response to the flashing K cursor, LET appears magically at the bottom of the screen. use it to type in:

```
LET x=2
```

and press Enter. All you get for your trouble is an OK message at the bottom of the screen and that's it. It's not too impressive until you try:

```
PRINT x
```

when:

2

appears. What's happened here? Previously when we tried:

```
PRINT x
```

we got an error message. Now, somehow, *x* stands for 2 and when we use PRINT on *x* we get the number 2 on screen. What's happened is that the:

```
LET x=2
```

has told the Spectrum that from now on the letter *x* will stand for the number 2. Anywhere an *x* is used in a PRINT statement – and in other statements as we'll see – the Spectrum knows that it stands for 2. Notice that it's:

```
PRINT x
```

Operator precedence

This is rather a grand name for what is, at this level, usually common sense. Don't worry if you don't follow it first time. It's one of those things that when you need to understand it, you will.

It should be fairly clear that:

```
PRINT 1+2+3
```

and

```
PRINT 3+2+1
```

will give the same results. The order of the figures and which plus is dealt with first doesn't matter when the signs (or operators, as they are called) are all pluses. The same applies to sums with all minuses or a mixture of plus or minus. The order of the operations isn't important. Try:

```
PRINT 2–3+6
PRINT 2+6–3
PRINT –1–2–3
PRINT –3–2–1
```

The same applies when the sum's operators are all multipliers or dividers or a mixture of both. Try:

```
PRINT 2*3*4,3*2*4
PRINT 64/2/4,64/4/2
PRINT 6*2/3,6/3*2
```

and you'll see that the operations can occur in any order. They give the same result.

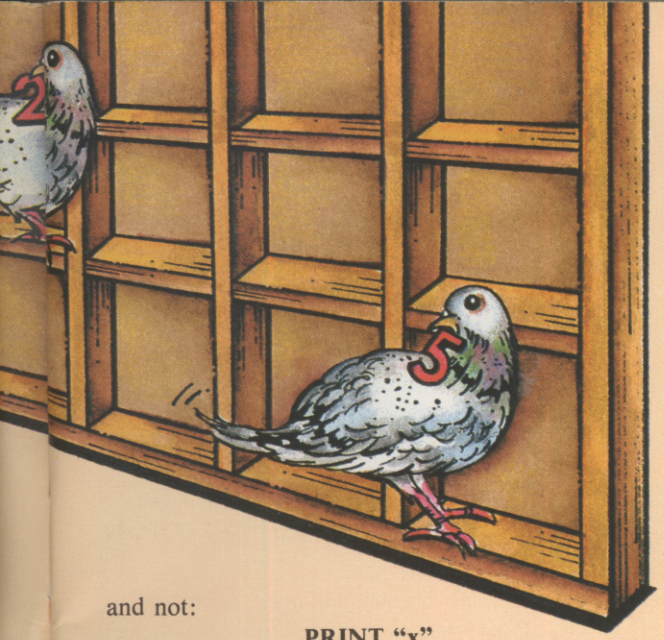
However when the plus/minus and multiply/divide operators are mixed up in one sum, or expression, then we've got problems. What if we are asked: "What is six plus eight multiplied by two?" Is it 14 (6 + 8) multiplied by 2, giving 28 as the result, or 6 plus 16 (2 × 8) giving 22? We get two different results depending on which operator is obeyed first. To sort this confusion out, we have to have a rule and it is:

*Starting from the left do anything in brackets first, **then** do all the divisions and multiplications, **then** do the additions and subtractions.*

To put it another way:

*( ) comes before * and /*
which come before + and –

This is the rule the Spectrum follows when it comes to evaluate expressions (solve sums) that contain a lot of operators (+, –, and so on). In fact it's the way we normally do our calculations without thinking about it. The trouble is that when we do start thinking about it the reasons for doing it this way seem a bit obscure! Take



and not:

PRINT "x"

The first uses *x* as a label for 2, the second just prints out what it finds between the quotes.

So, when your Spectrum comes across a statement like:

PRINT 2*x, x+4, x/2, x–x

it has no problems, provided that a previous LET has told it what *x* stands for. It just uses 2 wherever it finds *x* and carries on with the sum as before. To be formal, the letter *x* is what is known as an identifier or variable name. The statement:

LET x=2

tells the Spectrum that you want to use a variable named *x* and that, for the moment at least, its value is 2. There are strict rules as to which variable names you can use, as shown in the panel on the right.

comfort in the fact that if you do your sums following the rule, you'll get the correct answers.

So applying this rule to six plus eight multiplied by two we have:

6 + 8 × 2

giving:

6 + 16

which gives the answer:

22

Get your Spectrum to confirm this with:

PRINT 6+8*2

If we'd have wanted six to be added to eight and then the result of that to be multiplied by two we should use brackets as in:

PRINT (6+8)*2

The part in brackets is done first, then the multiplication.

Try using the rules to solve the expressions:

4×8+8/2+2
4×(8+8/2)+2
4×(8+8)/(2+2)
3×–7+21×2/7
1+2×2+3/3–3

and get your Spectrum to check your results.

So, by using the keyword LET we can get the Spectrum to give a number a label which we can use later. Why we should do this will become clearer as we learn more about programs and programming.

Variables vary

Let's set up a variable called *number* with:

LET number=3

So now, whenever I use *number* the Spectrum will realise that it's a variable name I'm using and substitute the number 3 in its place. Try:

PRINT number, number*2, number–1

and you'll get the answer

3 6
2

showing that the Spectrum has indeed used 3 wherever the *number* appeared. Now try:

LET number=4

followed by:

PRINT number, number*2, number–1

The result is:

4 8
3

as now the micro has substituted 4 wherever it came

Variable names

The rules for numeric variables (variables that stand for numbers) are simple but have to be obeyed. A variable name can be made up of any number of letters and characters, provided that it begins with a letter. Punctuation marks and operators are not allowed. So:

number
sum1

are acceptable variable names but:

1sttime
first*second

aren't, as you'll find if you try to use them in a LET statement.

You can, if you want, have spaces in your variable names, though the micro ignores them. So:

LET hours worked=10

is allowable and both:

PRINT hours worked

and

PRINT hoursworked

give the same result.

Finally, the Spectrum makes no distinction between upper and lower case in variable names. This means that as far as the micro is concerned the variable *NUMBER* is the same as the variable *number* which is the same as the variable *NuMbEr*.

One way of looking at variables is to see them as labels attached to boxes or pigeon holes in the Spectrum's memory. The command:

LET hours=20

does two things. It sets up a box labelled *hours* and puts the number 20 in it. Now whenever the Spectrum is asked to do something using *hours* it looks in the box labelled *hours* and asks the value it finds there.

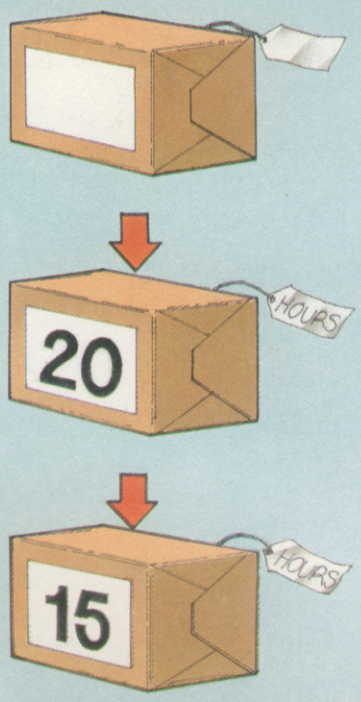
If now you enter another value for hours, say:

LET hours=15

then the box labelled *hours* now contains the value 15. The number that used to be there is there no more. The second LET has overwritten it.

Now when the Spectrum comes across an *hours* it goes to the box and uses the number it finds there, 15.

The label is still the same, and it's labelling the same box. What's changed is the value in the box. And that is how variables vary. The name stays the same, the value it stands for can change.



across *number*. The second LET has overruled the first one. The variable *number* was 3 but it's now 4. As you can see, the value given to a variable name by a LET isn't a once and for all thing. It can vary as later LETs give it a new value. And that is why they're called variables. The name stays the same, the value it represents varies.

These variables are powerful things. Try:

```
LET number=5
PRINT number,number+2, number-1
```

followed by:

```
LET number=6
PRINT number, number+2, number-1
```

and:

```
LET number=99
PRINT number,number+2, number-1
```

Each time we get different results but each time the print list:

```
PRINT number,number+2, number-1
```

stays the same. By the simple task of using LET to vary the value of *number* the same print list can produce lots of different results.

If you find the concept of variable strange it shouldn't be. We use them all the time. If I tell you that I'll pay you £2 for each hour you work then you know that your pay will be two times the number of hours you worked or, more formally:

```
pay=2*hours
```

If you work 10 hours you'll expect £20, if you work 15, £30. As the hours vary, so does your pay. You're using *hours* as a variable. It's a value that can change. One week it may be 5, the next 40 but whatever it is you still know that by using that value in:

```
pay=2*hours
```

you'll get the right answer. The variable name is used to show how the pay is worked out. When the variable gets a value then the pay can be worked out.

A first program

Let's put what we've learnt about PRINT, LET and variables together and create a program. Type in the following four lines, remembering to press Enter after each one.

```
10 LET first=1
20 LET second=2
30 LET sum=first+second
40 PRINT sum
```

If you find you've made a mistake in a line that you've entered, just retype the whole line again. On pressing Enter, the new, correct line will replace the old one.

The first thing you should notice is that instead of starting with a keyword, as was the case in our previous examples, these lines begin with a number. The next thing is that when they are typed in, nothing happens. Entering:

```
LET first=1
```

at least gives us an OK message. All we get for:

```
10 LET first=1
```

is a copy of the line on the screen. It's the same with the other three lines. What's happening?

The rule is that when a statement is entered just by itself, the Spectrum obeys it immediately. When it starts with a number in the range 1 to 9999, the micro doesn't obey the line but keeps it in memory as it knows it is going to use it later. So, as well as appearing on the screen, a copy of:

```
10 LET first=1
20 LET second=2
30 LET sum=first+second
40 PRINT sum
```

Editing programs

When you're entering programs, you'll often find typing errors creep in. What do you do if you wanted to enter the line:

10 PRINT 2*variable

and you get something like:

10 PRINT 2*varable

which you obviously don't want?

If you spot your mistake before you press Enter, then all you have to do is to use Caps Shift and the cursor control keys (5 and 8) in the usual way. While the line is still at the bottom of the screen it can be modified using the techniques already covered.

If you've already hit Enter then things are different. There's nothing on the bottom line to modify. The line sits there at the top of the screen in all its incorrect glory. One course of action is to just type in the whole line again. The new line automatically replaces the old line. So, if you had:

10 PRINT 2*varable

all you have to do to correct it is type in:

10 PRINT 2*variable

correctly.

The trouble with this is that all your previous typing is wasted. You have to type the whole line over again. It would be better if somehow the offending line could be made to reappear at the bottom of the screen. Then it could be modified in the normal manner. The Spectrum allows this to be done quiet easily.

Whenever a line is typed in there is always a chevron, >, between the number and keyword of that line. So entering:

10 PRINT "Hello Mum"

and pressing Enter, results in:

10>PRINT "Hello Mum"

appearing on the screen. Adding a new line such as:

20 PRINT "Hello Dad"

has the top of the screen becoming:

10 PRINT "Hello Mum"
20>PRINT "Hello Dad"

Notice that the chevron now points to the latest line.

It we've made a mistake and want to edit the line with the chevron, all we have to do is press the Caps Shift key and the 1 key at the same time. The result is that a copy of the marked line appears at the bottom of the screen. There it can be edited in the normal way. When it's corrected, pressing Enter sends it back into the micro, overwriting the old, incorrect line.

However it s not always the last line we entered that wants correcting. Often we notice that it is an earlier one. The remedy is simple. The chevron can be moved up or down the listing using the Caps Shift and 6 or 7 keys. Once the chevron is beside the line with the mistake, Caps Shift and 1 transfer it to the bottom line to be edited as normal. Once the line is modified, pressing Enter sends it back to the micro, replacing the incorrect line.

is held inside the micro.

These numbered lines make up a computer program. When the Spectrum is told to run the program, it looks for the line with the lowest number and obeys the statement following that. Once that is done it looks for the next lowest line number and obeys that. It carries on, obeying each line in sequence, until it runs out of lines to obey. Then the program stops. This is all a program is: An ordered set of instructions that the micro obeys, each one in turn.

Running a program

If you haven't already typed in:

10 LET first=1
20 LET second=2
30 LET sum=first+second
40 PRINT sum

and so got a program stored in the micro's memory, do it now. Then get the Spectrum to execute the program by using the R key to get the keyword RUN at the bottom of the screen and pressing Enter.

The result is that lines on the screen disappear and a 3 (the result of 1+2) appears. The message at the bottom of the screen just tells you that everything worked properly. But how did the program work?

When the Spectrum is told to RUN something it looks in its memory for the lowest line number and executes that statement first. In this case the lowest line number is 10 so the micro obeys the statement:

LET first=1

giving the numeric variable *first* the value 1. It then looks for the next line number and finds it is 20 so it goes on to processing:

LET second=2

setting up another variable *second* and giving it the value 2.

The micro next obeys line 30, adding the values found in *first* and *second* and storing them in *sum*. It goes on to line 40 and prints out the value of *sum* and then, running out of line numbers, the program stops with an OK message at the bottom of the screen.

It may not be the world's most stunning bit of Basic, but it is a program, a series of numbered statements, or program lines, that are processed in line number sequence by the Spectrum. If you want to see the program lines, or listing as it is called, then just press Enter after the program has finished running. Alternatively you can use the keyword LIST, found on the K key, to display a program listing. We'll be writing some more programs in Part III, when we meet a new Basic keyword, INPUT.

Discovering discs

We help you decide on the disc filing system that's right for you

EARLY on in the development of computers it became obvious that tape storage had an inherent drawback. No matter how fast the data could be made to transfer to and from the magnetic surface of the tape, there was always going to be a lot of time wasted in winding through the tape to find the particular piece of data you were after. This is called serial access, and it is now only used on large computers for security back-up copies, although different priorities – that is low cost – means it is usually the standard storage on many home computers, the Spectrum being no exception.

However, by taking the same kind of magnetic coating material that is used on tape and spreading it onto a disc, a recording surface with as much storage capacity as a piece of tape can be made. If one then arranges the recording tracks in concentric rings on the surface of the disc, very quick access to any part of the data can be had by skating the record/playback head across the surface to the desired track. This is called random access: You go straight to the data you want.

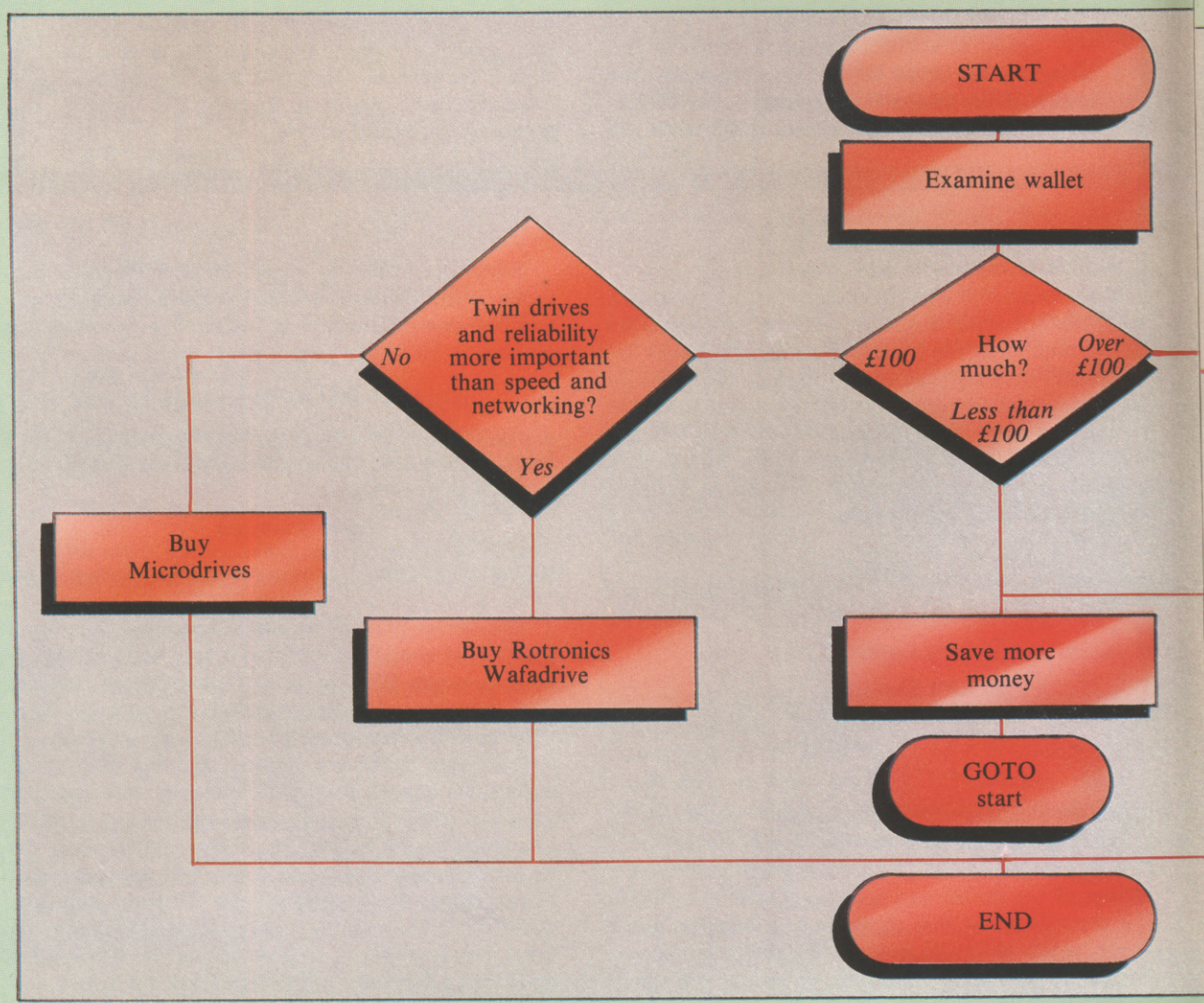
There is still an element of serial access while you wait for the section of the track you want to spin around to the head, but it is much more like random access, and discs are sometimes said to be random access.

The difference between the two methods can be understood easily in terms of music recording if you compare a long-playing record with a cassette recording of the same music. It is much quicker to find and start to play an individual track on the disc than it is on the tape.

Of course the computer disc drive cannot find its way around the magnetic disc by looking for the little gaps between the recordings like we do with LPs. However it knows exactly where each track is, and has the equivalent of an album cover to tell it what is to be found in each track. This is the disc directory, and it is recorded on one track of the disc itself so it can be constantly updated as the data stored on the disc is changed.

The first disc drives had stacks of discs made of rigid material and eight or even 12 heads all moving together so that they could read bytes of data in a parallel fashion instead of one bit at a time. Mainframes still use this sort of thing, but for other computers the size and cost of discs have been dwindling constantly.

Discs made of thin flexible material with a protective



sleeve were developed which could be removed from the drive, replaced with another disc holding a different set of programs, or even sent through the post. These "floppy" discs hold less information and are scanned by only one or two recording heads, so the transfer of data is slower. However their lower cost makes them more practical than the permanently installed hard platters for the general run of small computers. Miniature hard discs are available even for personal computers, but they cost a good bit more.

Smaller discs means less magnetic surface to record upon, but technical advances in recording materials and techniques has partly made up for this, and even the new shirt-pocket-sized discs can now be made to hold up to half a magabyte.

A few years ago no serious computer user would contemplate committing his data to anything less than an eight inch floppy, but now there is even an IBM PC with 3½in discs.

What's available?

A GOOD many add-on disc systems have become available for the Spectrum in the nearly four years that the computer has been around. Some that appeared have subsequently disappeared. At least one was advertised for months in full page colour magazine spreads complete with price list and order form without ever actually appearing. However the ones we list for you have been around for a while. Some of them are even beginning to get some software support!

None of the drives available are compatible with each other, in fact, they mostly don't even use the same type of disc. This is a drawback in terms of software availability, as manufacturers find it inconvenient to supply their products on a lot of different media. Transferring software from tape to disc is seldom straightforward, as the publisher will often go to great lengths to protect against copying, even though they will not provide their product on disc themselves.

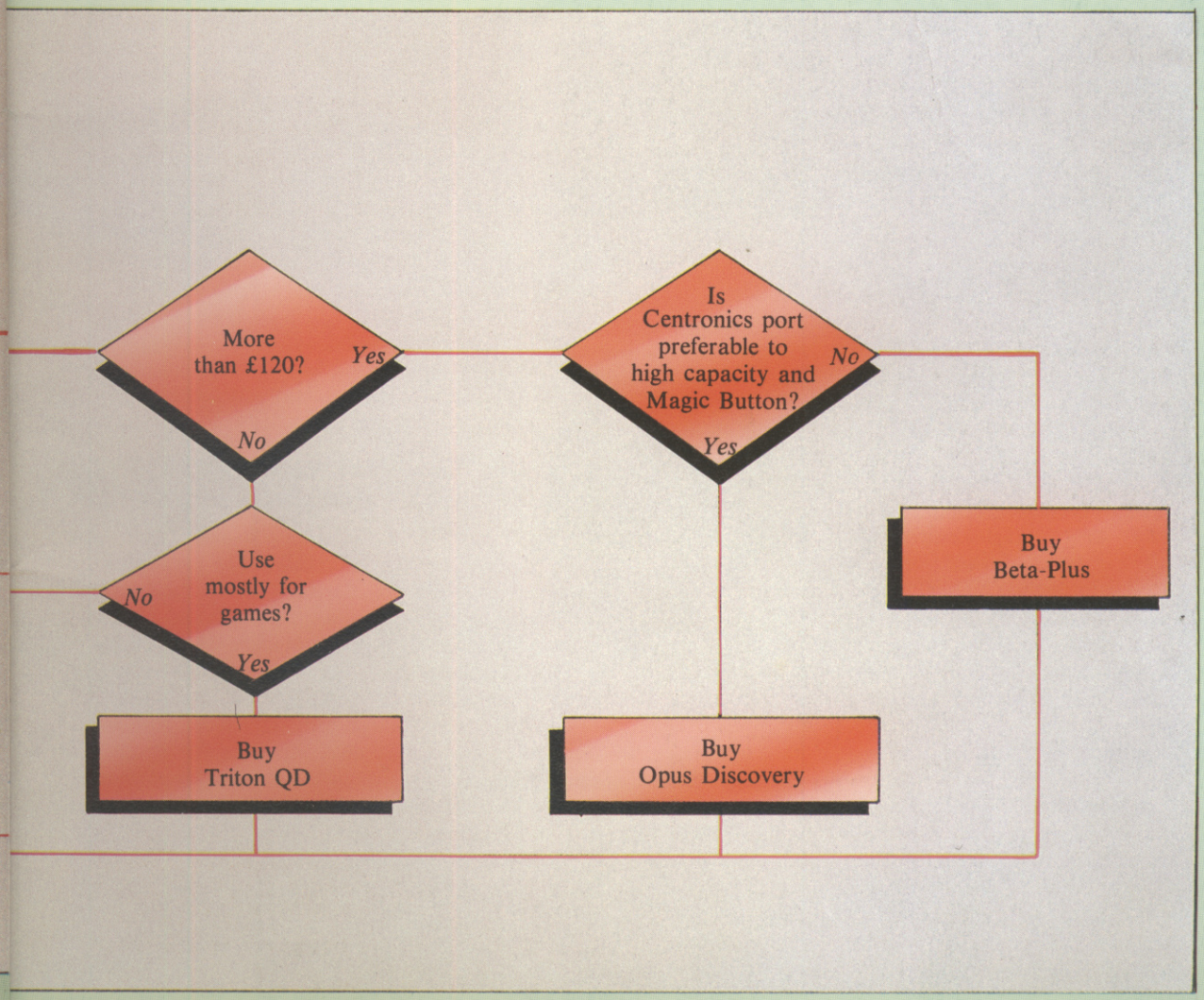
Most people buy a disc drive for rather more serious purposes than loading arcade games at a higher speed. Such software support as you find for discs, therefore, usually comes from the producers of serious utility software.

Other compatibility problems can arise when using two items with "shadow" ROMs on the same computer. Most discs will clash with microdrives, for instance, and the microdrives themselves sometimes clash with the VTX5000 Prestel modem. If you plan to use complex stacks of equipment it is wise to check compatibility with the supplier before you buy. Sometimes a modification is available.

Opus Discovery One

This is a complete upgrade all in one box. Besides a disc drive you get a Centronics parallel printer port, a Kempston compatible joystick interface (buy the stick separately) and a monochrome video output. The disc is the widely-used Sony designed 3½in type, and there is a space to have a second disc fitted in the largish Opus box. The edge connector is carried through for the use of other add-on peripherals.

There are a few minor drawbacks. The printer port uses an edge connector rather than a proper amphenol



socket, the video output is monochrome, not colour, and the box is designed to fit the Spectrum rather too snugly – fine with either the original Spectrum or the Plus, but the access is too cramped for most upgraded keyboards. A ribbon cable extender for the edge connector may be necessary.

The Discovery provides 170k of formatted storage per disc, which is a reasonable sized filing cabinet, almost twice the size of a microdrive cartridge. The discs themselves are more widely available than the other sub-five-inch media, and software is being made available on Opus format disc now.

The operating system is in a shadow ROM working in much the same way as the microdrive Interface 1. Opus has used the same command syntax as microdrives in an effort to be compatible with software that is designed to use microdrive. This does not work unless the commands are given in Basic. Machine code cleverness used in some microdrive-compatible software will fail on Opus discs.

Commands can be simplified in Opus syntax to save the typing fingers. It does this by defaulting to the *m* or drive channel, unless a different channel. like RS232, is specified. This means that the specifier and associated punctuation can be omitted from most commands.

Technology Research Beta-Plus

The Beta-Plus is so named because it is an improved version of the original Beta disc. This disc interface allows you to connect a number of different BBC-type disc drives to the Spectrum. This does not mean that you can share files with the BBC Micro, only that the drives are readily available. You can have one to four drives, which may be 5¼ or 3½in, single or double sided, and 40 or 80 track, to provide megabytes of on-line storage.

The Plus in Beta-Plus consists of an updated disc operating system (DOS) in ROM (well, eprom actually) and a Magic Button which allows you to transfer cassette-based software to disc without getting involved in protection busting. Some would say that the Magic Button does the protection busting for you, but this device is not a gift to the pirates, as the disc copy cannot easily be converted to a pirate tape, and you need the disc operating system to run the saved program.

Pressing the Magic Button saves an instant snapshot of the whole of the Spectrum's RAM onto disc. This may be done at any point in the running of a program, so it can be used to store a partially-completed word processing session or an adventure game in mid flow, to be taken up on the following day right where you left off. A drawback for the clumsy is the fact that this save takes place whenever the button is pressed, overwriting any priceless program or data already stored on the disc. Backup, backup, backup!

The Beta-Plus does not use the microdrive syntax, but something very like the cassette commands. Instead of just extending the Basic to give extra commands, the disc operating system is quite separate and you switch between Basic and DOS, depending on which functions you are using. Sometimes this switching is done automatically, as when you load an auto-run program from disc, and sometimes you do it by entering a command.

There is a detailed catalogue with complete information on every file on a disc, and you can rename files. The DOS uses the printer buffer for workspace and restores the contents after use, so there is no memory overhead to make transferring programs from tape difficult.

This is the only disc interface I know of that will co-exist with microdrives should you wish to use Interface 1 for its network or RS232 ports. If you want to use it with the VTX5000, ask for the modified version.

Triton QD

The Triton QD (Quick Disc) is a cheaper unit than the Opus or Beta, but it has lower capacity and no extra ports. It is possible to have two drives just by buying a second Triton and chaining the edge connectors.

The Quick Disc is a smaller disc than the Sony type (2.8 inches) and not as widely used, hence it is less easy to find the media in the shops. The drive has a flip up door like a cassette unit, and like a cassette, the disc can be turned over to play the other side. This means that of the 100k capacity given by each disc, only 50k is available as a filing cabinet at any one time. This is just big enough for the largest Spectrum programs, but too small to be practical for filing systems.

One advantage of the Triton drive is that its catalogue function gives a lot of extra information about file type and size. Another plus is that the disc operating system doesn't move the start of the Basic program area the way that the microdrives and some other discs do (it uses the printer buffer for workspace instead). This means that there are no memory problems when transferring a large program from cassette to disc. This disc was very fast on test.

The Triton operating system is, again, in a shadow ROM, but the command syntax is different from microdrive, without being an improvement. It is no easier to type and some commands cannot be used from within a program.

Rotronics Wafadrive

The Wafadrive is not strictly a disc, but a tape spooler similar to the microdrive. The cartridges are larger and more robust then microdrive cartridges, and they come in three capacities, 16k, 64k, and 128k, the smaller capacity sizes giving quicker loading times.

The Wafadrive is slower than microdrive, and it needs a 2k workspace – four times as much – which makes transferring large programs from tape more difficult, maybe impossible. However for about the same price as an Interface 1 and a single microdrive you do get two drives as standard, and the Wafadrive has both Centronics and RS232 printer interfaces fitted. You also get a free word processor thrown in, which is important, as other word processors won't be set up to use Wafadrives.

Wafadrive uses microdrive syntax, but allows a short form by defaulting to the drives, like the Opus disc.

Microdrives: What are they?

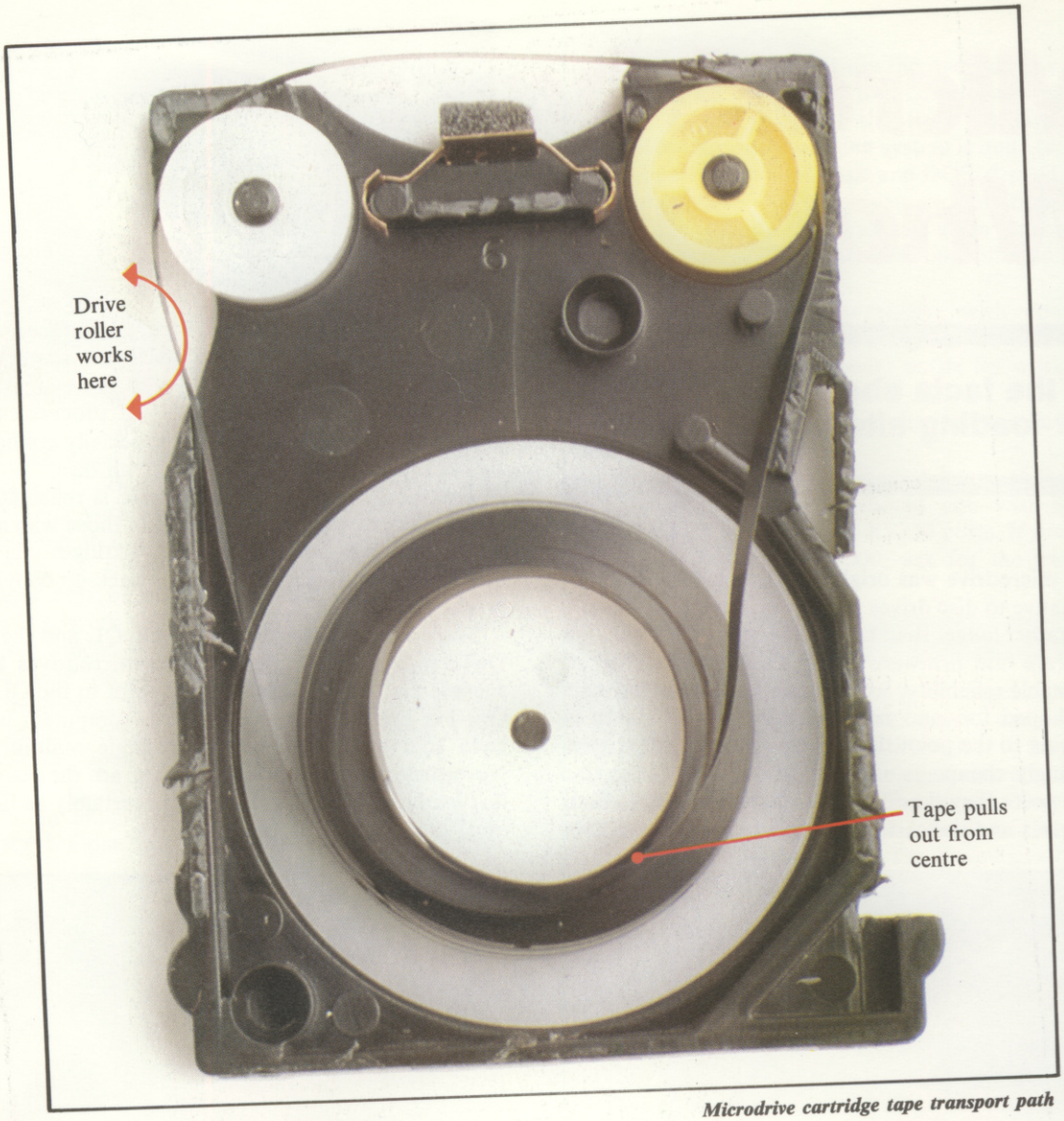
All the facts about this low-cost, fast-loading alternative to discs

THE microdrive was originally conceived as a cheap alternative to disc drives. Unfortunately for Sinclair, it took rather longer than had been expected for both the hardware and firmware to be brought to a point of reasonable reliability, so they didn't get quite the start they hoped for, and now the price of disc drives is dropping to the point that microdrives no longer look amazingly cheap.

However they do offer advantages over discs. Both the drives and the cartridges are much more compact than disc equipment. There are extra interfaces on the microdrive equipment for the Spectrum network and RS232 printers. And, possibly the most significant point, some software such as word processors and programming languages may be especially configured to work with microdrive but not with disc.

The most important disadvantage is reliability, or rather a degree of unreliability. Even those who swear by microdrives admit that the cartridges must be handled carefully and that regular back-up copying is essential.

There are those who point to the QL and the ICL One-Per-Desk as indications that microdrives are a success and here to stay. Others point to the QL and One-Per-Desk and say that these computers would have sold much better with a single built-in disc. Nevertheless, for the Spectrum they are the standard fast-loading upgrade, and they are certainly as fast as most discs, indeed faster than some.



Microdrive cartridge tape transport path

The hardware

To use microdrives you must have the Sinclair Interface 1, which slots on to the Spectrum edge connector and pretty well hides under the computer. The edge connector is carried through for other add-ons to be stacked. Other connectors are provided, including a pair of 3.5mm jack sockets for the Spectrum network, and a non-standard RS232 socket. The microdrives themselves are attached by means of a second, smaller edge connector on the left hand side. Up to eight drives may be neatly stacked together and plugged into this connector, but one or two are usually enough for most of us.

The drives are tape spoolers that drive an endless loop of tape at high speed past what appears to be a normal cassette tape head. The tiny microdrive cartridges – 3.5 by 4.5cm – contain about 11 feet of tape on a single spool, which feeds through by pulling out from the middle of the spool and winding back on to the outside. This method of tape handling imposes more stress and wear on the tape than a normal audio cassette, so the tape used is the more robust material used in video cassettes.

The firmware

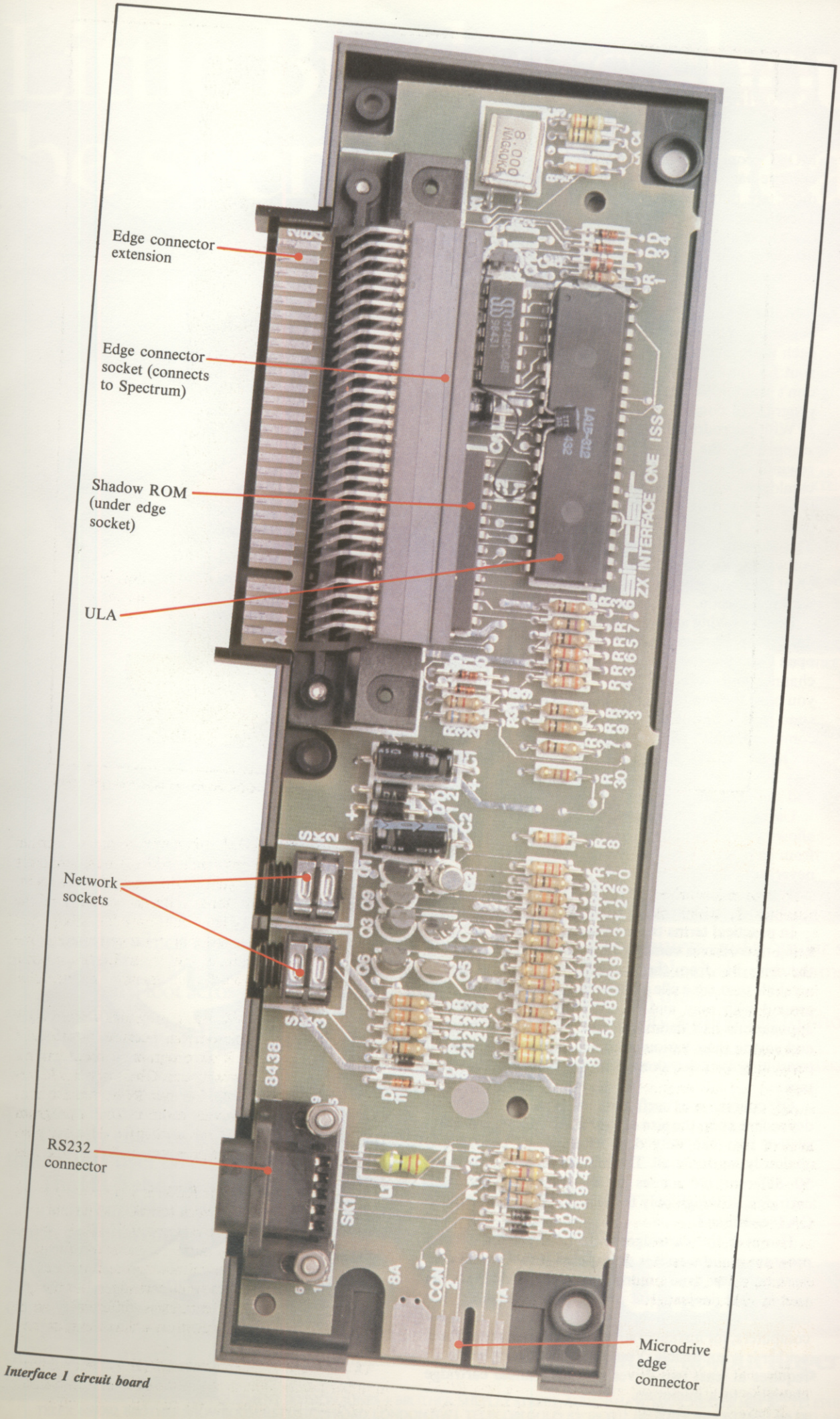
The extra commands used by the microdrive, network and RS232 port are handled by a new ROM in Interface 1. There is a circuit in Interface 1 that switches in this ROM whenever it senses certain addresses on the Spectrum's address bus, smoothly taking over from the main ROM inside the Spectrum.

There are at least three different versions of the Interface 1 ROM. The later ones have fewer bugs and run the drives a touch faster on some commands. One major bug does remain. If you try to LOAD a Basic program that is too big for the memory the machine crashes.

This can happen with Basic programs written on a Spectrum without microdrives because Interface 1 moves the start of the Basic program up about half a k to make room for its workspace. Obviously this leaves less room for programs, but the ROM routine that checks that there is enough room to load a program does not take account of the workspace. This can also occur with smaller Basic programs if RAMTOP has been moved.

The latest version of the ROM is supposed to work better when using the network to link a Spectrum to a QL, but there are so many versions of the QL ROM as well that it is hard to say anything about this for sure. The QL can use Spectrum microdrives as extra add-on drives, and it uses the same tape cartridges, but the QL apparently formats the cartridges differently, so it cannot read cartridges written on a Spectrum, or vice versa.

Unfortunately, the Interface 1 ROMs come soldered in rather than socketed, so you won't be able to just slot in the latest one, even if you can find a supplier.



Interface 1 circuit board

Storage and retrieval of data with the microdrive filing system

MOST people, when they contemplate upgrading from cassette-based computing to microdrives, or even discs, are thinking primarily of an increase in speed. There is another advantage, though. Microdrives and discs have a filing system which provides much greater flexibility and convenience in the way you store and retrieve software and data.

With cassettes you have to start and stop the tape recorder manually and rewind it to get back to the beginning. You have either to keep just one program on each tape, and sort out the one you want by hand, or wait for the tape to play through programs that you don't want while the computer watches for the right program to come under the head of the tape.

With microdrives and discs the drive is started and stopped by the computer, and there is no need to rewind. The filing system keeps a directory or catalogue which tells the computer where various programs and data are stored on the media, so it can pick out a program by name without a long search. This is particularly handy when you are using serious software like a word processor, which typically does a great deal of storing and retrieval during a single working session.

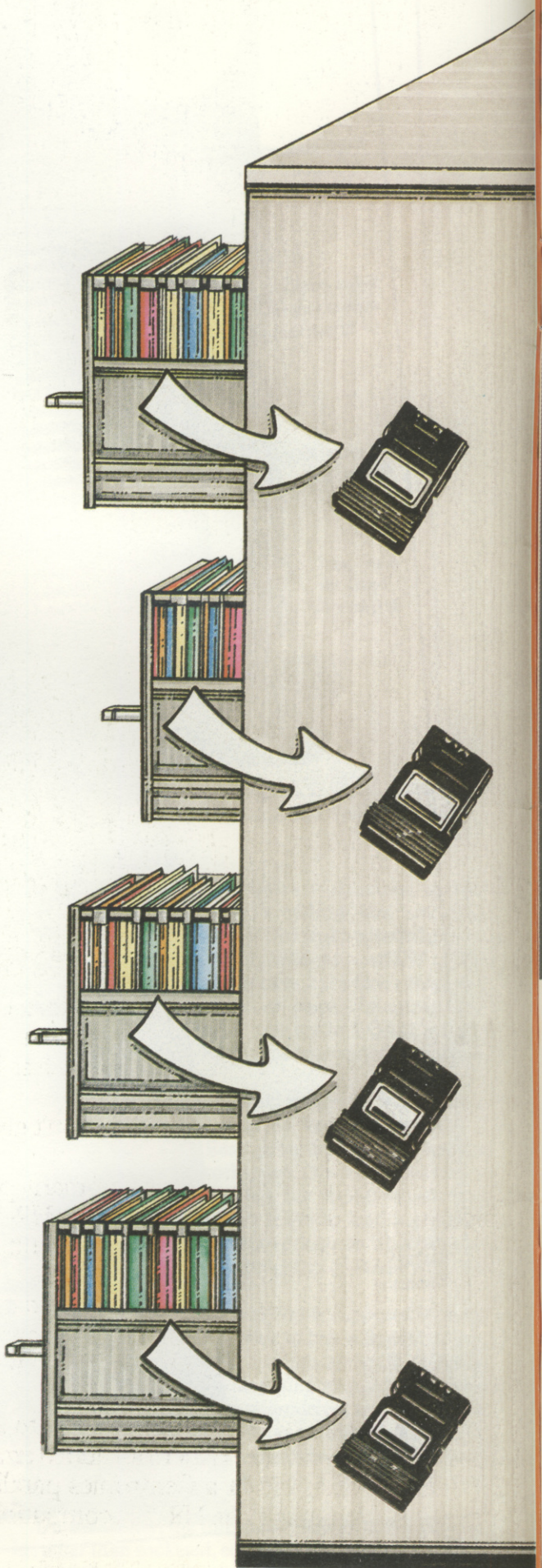
Besides saving and loading programs, you can store data from a program on microdrive. To do this you open a channel to the microdrive, print the data to that channel, and close the channel. To read the data later you open the channel and input the data. You can also send one channel to another with the MOVE command, which allows you to send a microdrive data file to the RS232 printer port, for instance. Channels may also be set up to move data between microdrives and the network.

Using the Spectrum channels is not simple, but it allows great flexibility in the handling of data output from the more serious type of software. You might never actually enter these commands directly, but if you use one of the word processor programs that works with microdrives, it may well employ them.

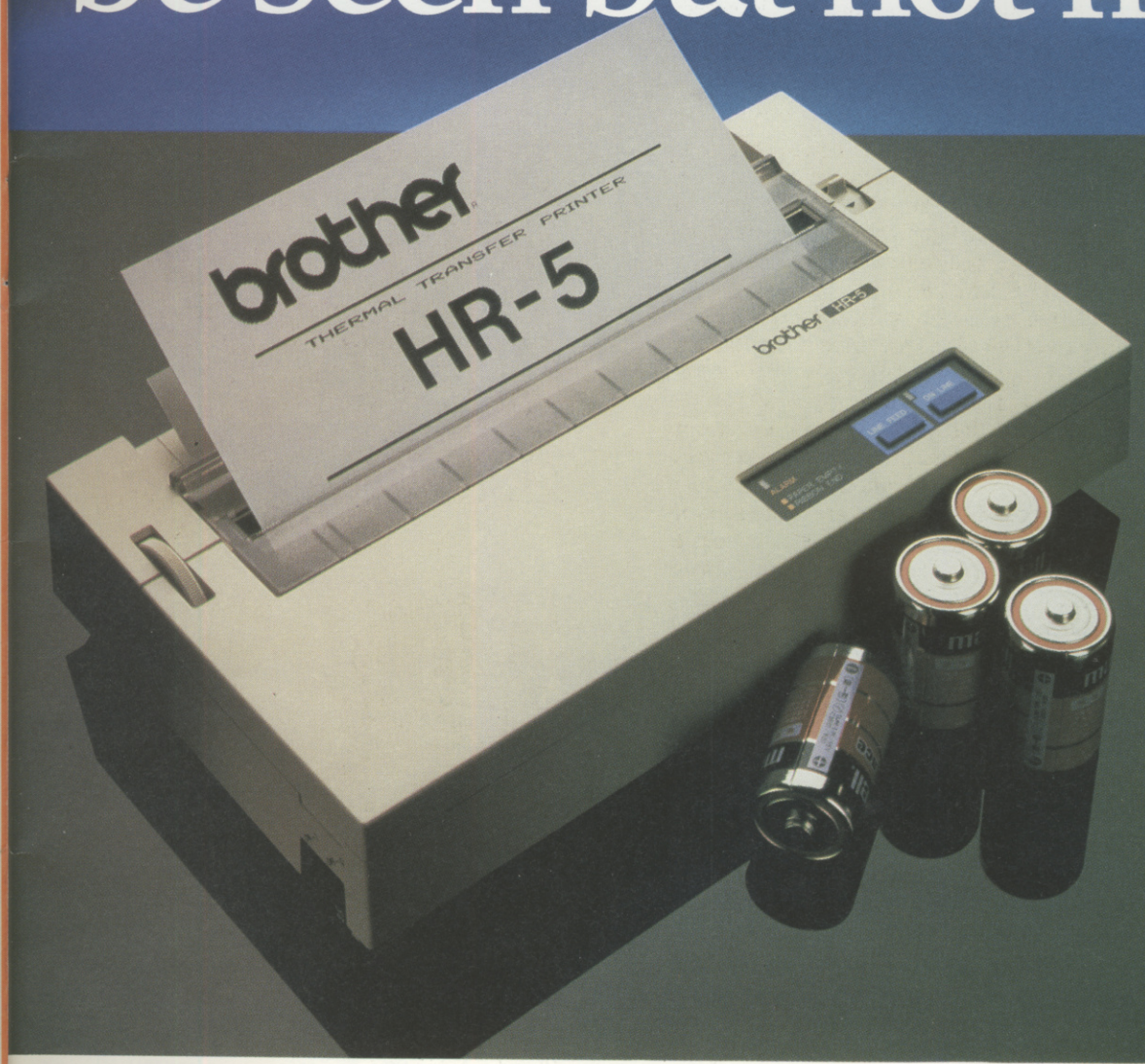
In practical terms the increase in speed gained from using microdrives comes not merely from the speed of the drive but from the fact that you need no longer scrabble through a pile of unmarked cassettes and mess around with play and record buttons. Sophisticated applications like databases can take care of loading and saving their various bits and pieces without even consulting you, let alone prompting you to rewind a tape.

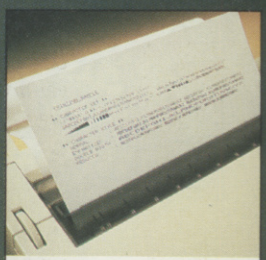
Of course, the usefulness of a filing system is dependent upon the size of the "filing cabinet". A filing system that can only deal with one or two files is obviously no use at all. The microdrive will allow over 50 different file names in the catalogue for each cartridge, although only the first 50 are displayed by a CAT command.

However the cartridges only hold around 90k of program or data, so it is doubtful that you would ever have occasion to use more than a dozen different file names. 90k is just 10k short of the smallest disc capacity in common use (on the Osborne or BBC computers, for instance) so while it is not commodious, neither is it impossible. You may find that serious work requires at least two drives to keep manual cartridge changing to a minimum.

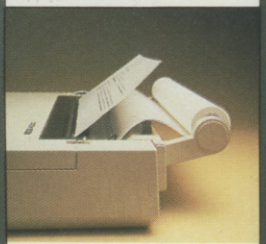


Little Brothers should be seen but not heard.

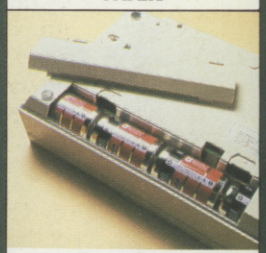




REGULAR, CONDENSED, OR EXTENDED FACES.



CUT SHEET A4 OR ROLLER PAPER.



BATTERY OR MAINS OPERATED.

A maxim which eloquently describes the Brother HR-5.

Less than a foot across, it's nonetheless loaded with features.

But there's one thing the HR-5 won't give you. Earache.

For the annoying 'clickety clack' many printers produce is mercifully absent from the HR-5.

Quietly efficient, it delivers high definition dot matrix text over 80 columns at 30 c.p.s.

The HR-5 also has something of an artistic bent.

Being capable of producing uni-directional graph and chart images together with bi-directional text.

It will also hone down characters into a condensed face, or extend them for added emphasis.

Incorporating either a Centronics parallel or RS-232C interface, the HR-5 is compatible with most home computers and popular software.

Perfectly portable, the battery or mains operated HR-5 weighs less than 4lbs.

Which is really something to shout about.

PLEASE SEND ME MORE DETAILS OF THE REMARKABLE BROTHER HR-5 PRINTER.

NAME _____

ADDRESS _____

_____

_____

_____ TEL NO. _____

brother®

The future at your fingertips.

DEPARTMENT P, BROTHER OFFICE EQUIPMENT DIVISION, JONES + BROTHER, SHEPLEY STREET, AUDENSHAW, MANCHESTER M34 5JD. TELEPHONE: 061-330 6531. TELEX: 669092. BROTHER SHOWROOM: 83 EUSTON ROAD, LONDON NW1 TELECOM GOLD: 83: JBC002. BROTHER INDUSTRIES LIMITED, NAGOYA, JAPAN.

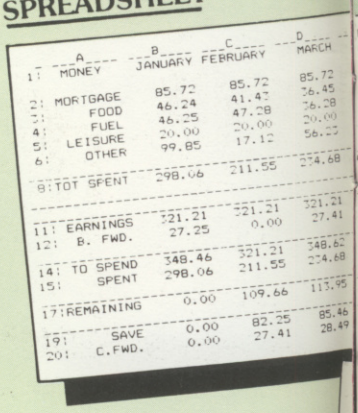
QUICK TO LEAR

THAT'S...

MINI OFFICE

FINALIST BRITISH MICRO COMPUTING AWARDS 1985

SPREADSHEET



	A	B JANUARY	C FEBRUARY	D MARCH
1:	MONEY			
2:	MORTGAGE	85.72	85.72	85.72
3:	FOOD	46.24	41.43	36.45
4:	FUEL	46.25	47.28	36.28
5:	LEISURE	20.00	20.00	20.00
6:	OTHER	99.85	17.12	56.23
8:	TOT SPENT	298.06	211.55	274.68
11:	EARNINGS	321.21	321.21	321.21
12:	B. FWD.	27.25	0.00	27.41
14:	TO SPEND	348.46	321.21	348.62
15:	SPENT	298.06	211.55	274.68
17:	REMAINING	0.00	109.66	113.95
19:	SAVE	0.00	82.25	85.46
20:	C.FWD.	0.00	27.41	28.49

JUST LOOK WHAT THIS PACKAGE CAN DO!

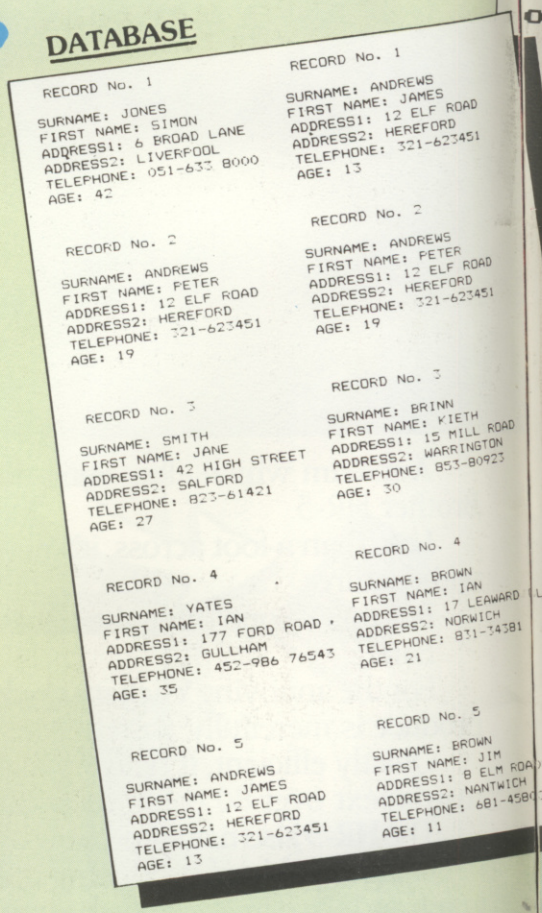
WORD PROCESSOR – Ideal for writing letters or reports! *Features:* Constant time display ● Constant word count (even shows words per minute) ● Normal or double-height text on screen or printout.

SPREADSHEET – Use your micro to manage your money! *Features:* Number display in rows and columns ● Continuous updating ● Update instantly reflected throughout spreadsheet ● Save results for future amendments.

GRAPHICS – Turn those numbers into an exciting visual display! *Features:* 3D bar chart ● Pie chart ● Graph.

DATABASE – Use it like an office filing cabinet! *Features:* Retrieve files at a keystroke ● Sort ● Replace ● Save ● Print ● Search.

DATABASE

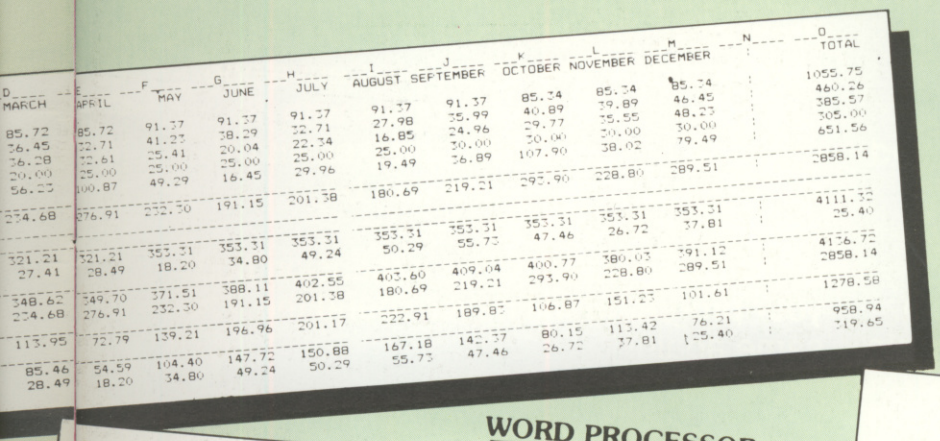


```
RECORD No. 1
SURNAME: JONES
FIRST NAME: SIMON
ADDRESS1: 6 BROAD LANE
ADDRESS2: LIVERPOOL
TELEPHONE: 051-633 8000
AGE: 42

RECORD No. 2
SURNAME: ANDREWS
FIRST NAME: PETER
ADDRESS1: 12 ELF ROAD
ADDRESS2: HEREFORD
TELEPHONE: 321-623451
AGE: 19

RECORD No. 3
SURNAME: SMITH
FIRST NAME: JANE
ADDRESS1: 42 HIGH STREET
ADDRESS2: SALFORD
TELEPHONE: 823-61421
AGE: 27

RECORD No. 4
SURNAME: YATES
FIRST NAME: IAN
ADDRESS1: 177 FORD ROAD
ADDRESS2: GULLHAM
TELEPHONE: 452-986 76543
AGE: 35

RECORD No. 5
SURNAME: ANDREWS
FIRST NAME: JAMES
ADDRESS1: 12 ELF ROAD
ADDRESS2: HEREFORD
TELEPHONE: 321-623451
AGE: 13

RECORD No. 1
SURNAME: ANDREWS
FIRST NAME: JAMES
ADDRESS1: 12 ELF ROAD
ADDRESS2: HEREFORD
TELEPHONE: 321-623451
AGE: 13

RECORD No. 2
SURNAME: ANDREWS
FIRST NAME: PETER
ADDRESS1: 12 ELF ROAD
ADDRESS2: HEREFORD
TELEPHONE: 321-623451
AGE: 19

RECORD No. 3
SURNAME: BRINN
FIRST NAME: KIETH
ADDRESS1: 15 MILL ROAD
ADDRESS2: WARRINGTON
TELEPHONE: 853-80923
AGE: 30

RECORD No. 4
SURNAME: BROWN
FIRST NAME: IAN
ADDRESS1: 17 LEAWARD
ADDRESS2: NORWICH
TELEPHONE: 831-74381
AGE: 21

RECORD No. 5
SURNAME: BROWN
FIRST NAME: JIM
ADDRESS1: 8 ELM ROAD
ADDRESS2: NANTWICH
TELEPHONE: 681-4580
AGE: 11
```

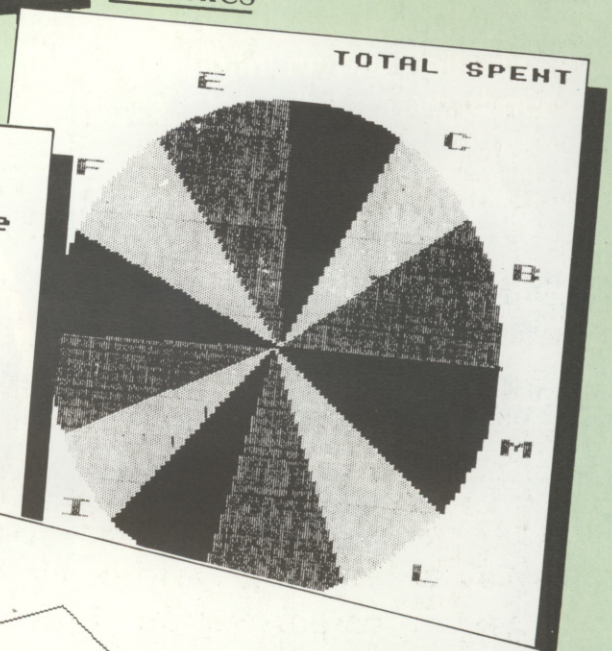
...and it's all at price of just

RN, EASY TO USE

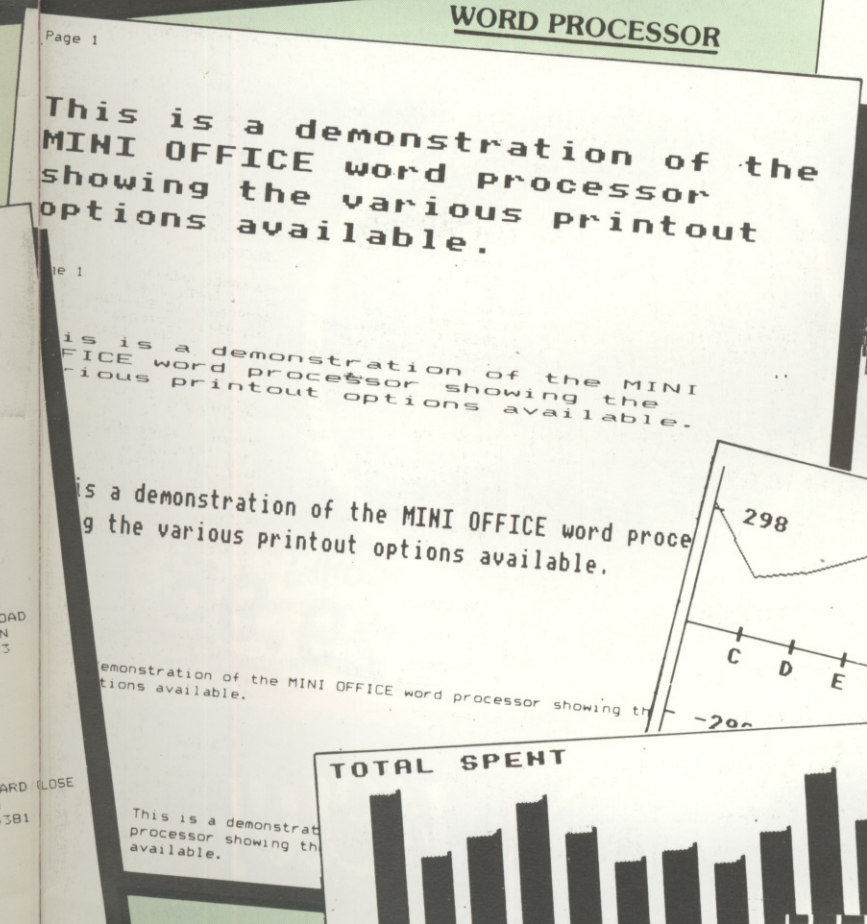
Spectrum 48k and Spectrum+

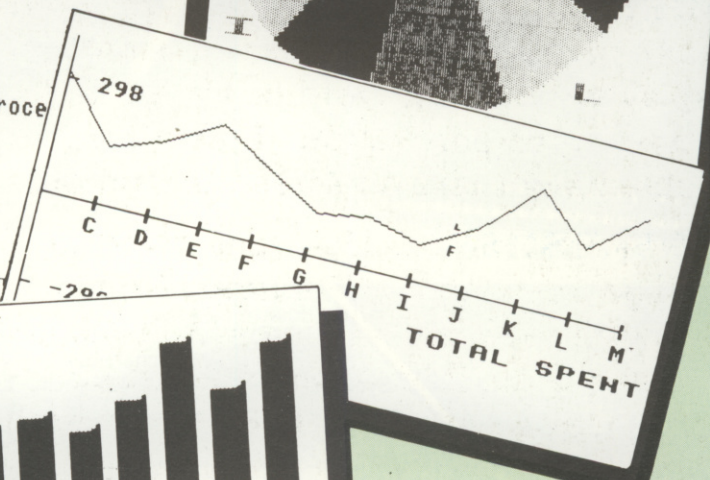


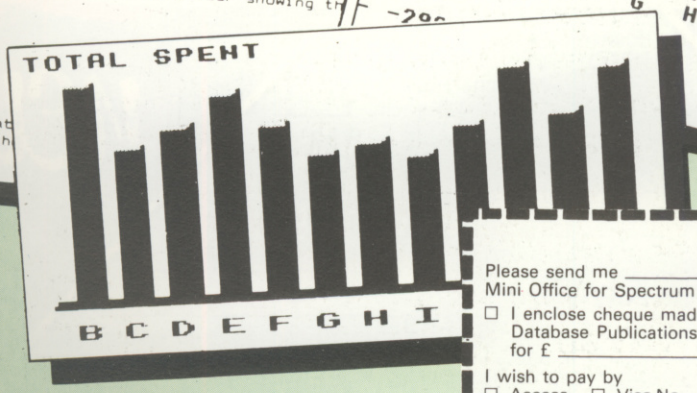
GRAPHICS



WORD PROCESSOR







Please send me _____ copy/copies of Mini Office for Spectrum 48k/Spectrum+

☐ I enclose cheque made payable to Database Publications Ltd. for £ _____

I wish to pay by
☐ Access ☐ Visa No. _____ Expiry date _____

Signed _____

Name _____

Address _____

Post to: Mini Office Offer, Database Publications, 68 Chester Road, Hazel Grove, Stockport SK7 5NY.

Only £5.95

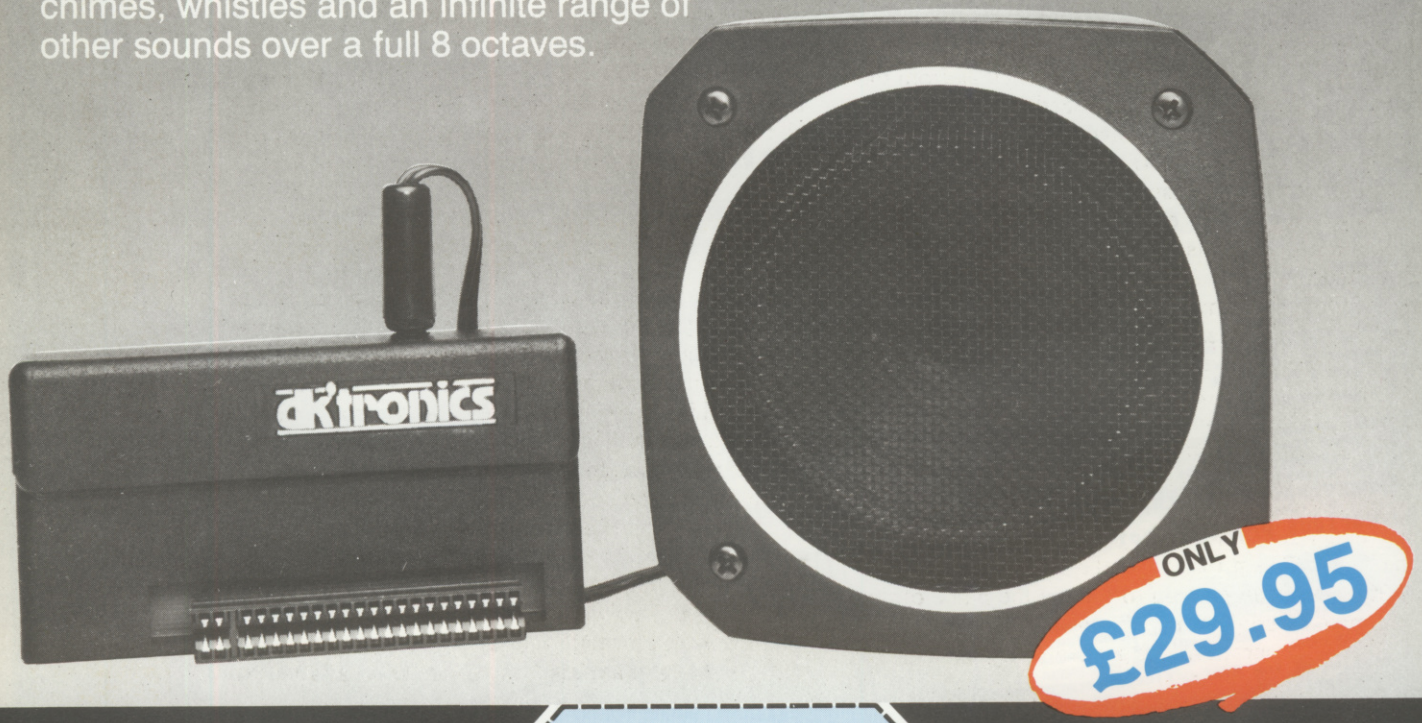
at the unbelievable
st £5.95 CASSETTE

DATABASE SOFTWARE

SOUND IDEAS FOR YOUR SPECTRUM

The Three Channel Sound Synthesiser interface incorporates a BEEP audio amplifier and a 3 channel sound synthesiser.
The BEEP amplifier improves the sound quality and output of the BEEP enormously. The 3 channel sound synthesiser adds a totally new dimension to sound on your Spectrum. It allows you to program your own music with harmonies, explosions, zaps, chimes, whistles and an infinite range of other sounds over a full 8 octaves.

Based around the popular AY-3-8912 sound chip it gives you complete control (from basic or M/C) over 3 channels of tone and/or white noise, plus envelope and volume control. It comes with its own pod mounted (4") speaker with 1 metre of cable so that it can be positioned anywhere.
Once this is fitted to the expansion port your programs will never sound the same again!



ONLY £29.95

It's Available NOW!

Please rush me the following

..... Three Channel Sound ..... Synthesiser Interfaces @ £29.95 each. £____
Please add post and packing £1.25
I enclose cheque/PO/Cash for ..... Total £____
or debit my Access/Barclaycard No.

Signature _____ Name _____

Address _____

Or send S.A.E. for the New D.K.Tronics Spectrum Catalogue

NEW!

"Available direct or from good computer shops anywhere"

dk'tronics

DK Tronics Ltd., Unit 6, Shire Hill Industrial Estate, Saffron Walden, Essex CB11 3AQ. Telephone: (0799) 26350 (24 hrs) 5 lines

The Spectrum Connection

Microdrives: The commands explained

Saving and Loading

SAVING and loading with microdrives obviously requires different commands from the ones used for the cassette recorder. There are a number of reasons why the syntax of the new commands has to be more complex than the old. For one thing, something must be added simply to distinguish the microdrives from the tape. Then the existence of the filing system means that you must use a program name when loading, as there are likely to be several programs on the cartridge, in no particular order.

Furthermore, you can LOAD and SAVE from the network port and the RS232 port, as well as from the drives, so more identifiers are needed to specify them. And finally there are extra functions associated with a file structure, such as formatting and cataloguing, which need their own instructions.

All the necessary extra bits, plus a deal of punctuation that seems quite unnecessary but cannot be omitted, makes the syntax of the new commands long winded and unwieldy. There is, however, one quick short cut that can be used to good effect. If you put a program with the name "run" on a cartridge, it can be loaded automatically just by giving the computer a RUN command. This only works if the RUN is the first command given after power-up or a NEW, but it is very useful all the same.

Many people write a little menu-driven loading utility and put it on to every cartridge, giving it the name "run". This program displays a catalogue of the cartridge and prompts you to type in the name of the program you want to load, without all the complicated bits. Such a program might go like this:

```
10 CAT 1
20 INPUT "NAME TO LOAD? ";a$
30 LOAD *"m";1;a$
40 STOP
100 SAVE *"m";1;"run" LINE 10
110 REM Line 100 used to SAVE this program
```

As you can see in line 100, when using microdrives there is a lot to type before you even get to the name of the program. The asterisk (*) is used to kick the Interface 1 "shadow" ROM into life. The "m" specifies microdrive rather than network, and so on. The 1 indicates that it is microdrive number one you are using (this is needed even if you only have one). The "LINE 10" part is exactly like the cassette command – it makes the program run from line 10 automatically when it loads.

The quotation marks and semicolons sprinkled through the syntax are only there because the Spectrum is using the same ROM subroutines to process microdrive commands as it uses for PRINT items, so you must include them even though they convey no essential information.

The special commands you'll need when you use microdrives

MERGE and VERIFY

You can MERGE or VERIFY Basic programs from microdrive in the same way as with cassette. However programs that have been saved to microdrive with a RUN-on-LOAD line number will not merge. This fact allows commercial programs to be protected from copying.

Formatting

When you buy a new blank microdrive cartridge the first thing you must do is FORMAT it. This process runs through the loop of tape, testing it and placing markers on it, and makes a map so that the computer knows which parts are usable and which it should avoid, such as the join where the loop is spliced. When formatting a cartridge, you can give the cartridge itself a name which will appear in the catalogue.

Some experts recommend that you FORMAT the cartridge several times on the trot when you first get it, to give it a good workout before committing any program to it. You will find that each time you format a cartridge it will format to a different capacity and you may wish to maximise the storage available by

formatting repeatedly until you get a high figure for storage capacity (use the CAT function to find out the capacity). You usually get about 90k of storage on each cartridge, and if a new one won't give you at least 85k it is substandard.

Formatting a cartridge erases it completely, so once you have put a program on a cartridge you must take care not to use FORMAT accidentally. You may think that it is an unlikely mistake to make, but it has happened to others!

The FORMAT command is also used to set the baud rate for the RS232 port and to assign station numbers for the network.

The Catalogue

Each cartridge has a catalogue kept on it which is automatically updated to include everything you save to the cartridge. The command CAT 1 will give you a list of the programs on the cartridge in drive 1, plus the name you gave to the cartridge itself, and the number of kilobytes of space left free.

The catalogue unfortunately does not tell you which names on the list are Basic programs and which are blocks of code or data. If you try to load a "program" which is actually a code block, and so on, you will get a "Wrong file type" error report, but it still won't tell you what the file actually is. It is also possible to save a file with an invisible name, which puts an even greater strain on the memory.

Erasing

There is no need for an ERASE command with cassette storage since you merely record over the old program, but you need a way to remove redundant files from the microdrive. Formatting wipes all the programs on a cartridge, but by using the ERASE command you can remove just the one named program.

This is particularly necessary when you are in the process of writing a program and saving partially-completed versions to microdrive from time to time. Unless you give each save a new name, which will quickly fill the cartridge, you will have to erase the old version before saving the new one. This is because the microdrive cannot have two programs in a catalogue with the same name. How would it be able to choose between them when loading?

Microdrive cartridges can be protected against accidental erasure by removal of the write protect tab. This is much the same as the record prevention tab on a cassette – it stops not only erasure and formatting, but any saving operation as well, so it is only of use for permanent program storage, not for changeable files of data.

Channels and Streams

The Extended Basic in the Interface 1 ROM has powerful facilities for directing the flow of data from one device to another, called channels and streams. Actually these are at work in the standard Spectrum as well, but they are transparent to the user – that is, you don't notice them. When the Interface 1 is fitted you can program the channels and streams to very useful purposes, and the people you buy your software from can use them to make life easier for you.

There is, as far as I know, no law of programming that states that the more powerful and flexible a technique is, the more difficult it will be to understand. Still, this is often the case, and so it is with the Extended Basic commands for channels and streams.

Standard Spectrum channels

k	The keyboard in input or the lower screen in output. (Obviously you can not output to the keyboard or input from the screen.)
s	The upper 22 lines of the screen, output only.
p	The ZX printer, again output only.

Interface 1 extends the number of channels, adding:

t	Text output via the RS232 port. Used for listing, doesn't send unprintable characters, and expands Basic keyword tokens into spelled out form. Works in input but ignores 8th bit. Must have baud rate set with FORMAT.
b	Binary data in and out via the RS232 port (full eight bits). The baud rate must have been set with FORMAT.
n	Network input and output, requires a station number.
m	Microdrive in and out, requires drive number and file name.

Figure I

Behind these fluvial names lurk even more complicated machine code data manipulation routines. The names are meant to simplify things by concentrating the mind on the function of the commands, rather than the intricacies of the computer's operations.

Channels are devices that data can be sent to or received from. Streams are the routes that the data moves along. These are logical devices and routes, not physical ones. Each channel is an input or output routine in the Spectrum or Interface 1 ROM, and the streams are really a way to assign labels so you can pick which channels you wish to use.

There are 16 possible streams in the Spectrum, but the first four are already assigned to link various channels used by the normal Basic commands. PRINT for instance uses stream#2, which is linked to the upper screen output channel. When you use PRINT#0 or PRINT #1, you divert the output via the streams ordinarily only used by the Basic editor, and the print appears in the lower two lines of the screen.

You can assign the streams from 4 to 15 to any of the channels by using the OPEN # command and CLOSE # will close it again. CLOSE # is important, especially with microdrive data files, as just printing to a file does not actually put the data on the cartridge until the file is closed. The channels are identified by single letters enclosed in quotes. These letters and the channels they represent are shown in Figure I.

By assigning a stream to the RS232 "t" channel and LISTing to that stream, you can LIST a program on a full size printer. Or you can PRINT to the stream. You can even send the catalogue down the stream with a CAT command, to print out the microdrive catalogue.

When saving and loading programs to the microdrive you don't have to open a stream, you just use an asterisk after the SAVE or LOAD, followed by the microdrive identification bits and pieces. However

handling a data file is made easier by using streams. You can open a stream to a file on the microdrive and then PRINT the data to the file via the stream.

The MOVE command

MOVE will move data from one channel to another, via the assigned streams. This is a very flexible command which can send a microdrive file to the screen, the network, or even another microdrive file. You could even use it to connect the keyboard directly to the RS232 printer port. MOVE will open and close the streams automatically, but it only works on data. To send program files whizzing around the place you must first load them and then SAVE or LIST them to the stream of your choice.

More information on how to use the various channels and streams associated with the drives, network and RS232, is in the manual that comes with the microdrives. Luckily, you won't need channels and so on to begin with, or maybe ever, because this part is not simple. A number of books have been written to try to make the microdrives easier to understand, but whatever you do, don't lose the manual.

What's missing?

Most of the facilities offered by disc operating systems are available in the microdrive's Extended Basic. The one that has been left out is a rename command. If you want to rename a program or file on microdrive, you have to save it under the new name and erase the file with the old name.

Cartridge care

The tape in microdrive cartridges is subject to a good deal of wear. There is nothing you can do about this except make sure that you don't lose your programs when the cartridge expires. You do this by making back-up copies. The back-ups don't have to be on cartridge, cassette tape will do. You aren't supposed to use them, just keep them safe to avert a disaster when a cartridge goes wrong. This advice applies to discs, too, but many people have reported more than usual reliability problems with microdrives, and extra care is warranted.

You must not turn the power to the computer on or off while a cartridge is inserted in the drive, nor remove a cartridge while the drive is running.

These actions are almost certain to corrupt at least part of what is stored on the cartridge and make it unreadable.

The cartridges are mechancially delicate and should be kept in their cases when not in use. Ashtrays, cups of tea, and the tips of magnetic screwdrivers all display a disturbing attraction for unprotected cartridges.

Microdrive command summary

CAT x — **9 key, Ext Mode**

Lists the files on the cartridge in microdrive number x. Use CAT # 3;x to list on ZX printer.

CLOSE #x — **5 key, Ext Mode**

Closes stream x. Sends any data left in buffer to file or network.

ERASE "m";x;"name" — **7 key, Ext Mode**

Erases file "name" from cartridge in microdrive x.

FORMAT "m";x;"name" — **0 key, Ext Mode**

Formats cartridge in drive x, and names it "name". This erases anything already stored on the cartridge. Every cartridge must be formatted before it can be used.

FORMAT "n";x — **0 key, Ext Mode**

Sets the network station number to x (range 0 to 64).

FORMAT "t";x — **0 key, Ext Mode**

Sets the baud rate for the RS232 text channel to x. Use "b" instead of "t" to set the binary channel. Must be standard baud rate: 50, 110, 300, 600, 1200, 2400, 4800, 9600, 19200.

INKEY$ #x — **N key, Ext Mode**

Returns a character from stream x if available.

INPUT #x;variable — **I key**

Inputs a variable (number or string) from stream x.

LOAD *channel;options — **J key**

Loads from channel "b", "n" or "m" (specify). Options are usual load options. Usual form is LOAD *"m";1;"name" for loading program "name" from microdrive number 1.

MERGE *channel;options — **T key, Ext Mode**

Like cassette MERGE but from channel specified.

MOVE x TO y — **6 key, Ext Mode**

Moves data from x to y, where x and y are either stream numbers of channel identifiers. Streams are automatically opened when x or y are channels. This command will not terminate until an end of file marker is reached, and this only happens when the source is the network or microdrive.

OPEN #x — **4 key, Ext Mode**

Opens stream x. Stream should not already be open.

PRINT #x;items — **P key**

Prints usual print items via stream x.

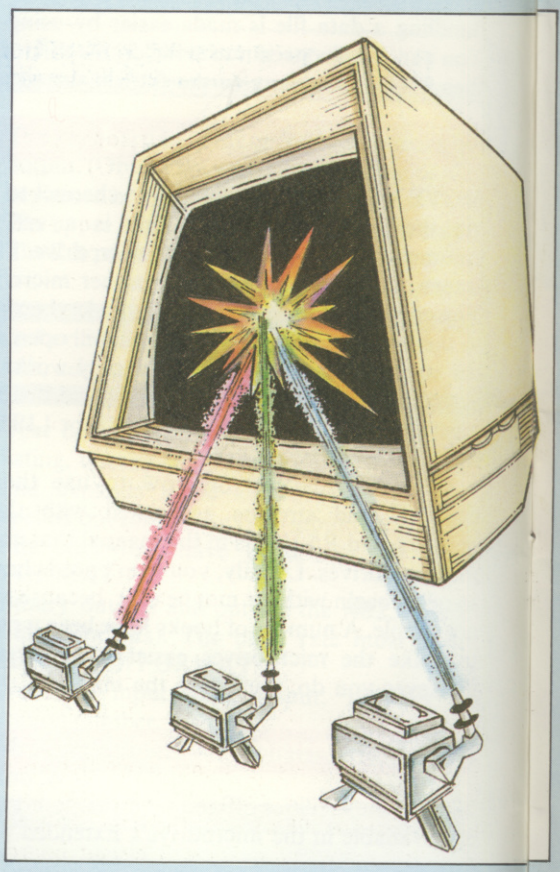
SAVE *channel;options — **S key**

As LOAD above, but saves.

VERIFY *channel;options — **R key, Ext Mode**

As LOAD above, but only compares file to check that previous SAVE was error free.

Monitoring the Video



Discover the benefits of using a colour monitor in place of the domestic TV set

THE Spectrum's TV modulator can put out a very good display on a colour TV set, particularly the later Spectrums from Issue 3 onwards. A change was made to the ULA in Issue 3 that made the colour output work better especially with some foreign makes of TV that had given trouble with Issues 1 and 2.

Not everyone is happy with the TV display, however, and it certainly can be improved with a colour monitor. There are two kinds of video connection to a colour monitor, and the Spectrum, as it comes, is not equipped with either one.

RGB

The better sort of monitor uses RGB signals, which stands for red, green, and blue. The Spectrum only has what are called "colour difference" signals available. These correspond to red minus yellow, blue minus yellow and luminance plus sync.

A special monitor is available for the Spectrum from Microvitec that will make sense of these signals, and very good it is too, although expensive. It attaches to the edge connector with no fuss, and gives an extremely sharp picture with no "dot crawl", the small scale shimmering you'll see. This is really a RGB type monitor with some extra circuitry to cope with colour difference signals.

Composite video

The other type of monitor uses a composite video signal. In composite video, all the colour signals have been coded together into one complex signal, so you need only one coaxial conductor to carry it around the place, which is why this is the type of video signal that the television companies themselves use in most of their monitoring equipment. There is a composite video signal available on the Spectrum board, but it may not be present at the edge connector.

From about Issue 3 onwards the composite video signal was brought out to the edge connector, where it can be found on the underside of the board, fourteenth pad from the non-slotted end of the connector. Earlier issues were supplied with an open link, which meant you had to open the case and make a join to get the circuit to carry out to the edge connector. Whatever issue you have though, using a whole edge connector socket just to make a single connection seems a little awkward, especially as there is an easier way.

Dashing away with a soldering iron

IF you have never, ever, before messed about with solder and veroboard, connecting dinky electronic gubbins together into oscillators that didn't oscillate or amplifiers that did, then perhaps you shouldn't read this.

A computer is about as complicated a piece of circuitry as comes, and it is not the ideal training ground for a keen learner with a new soldering iron. So if you're still with me, I'll assume you know how to solder. It can't be taught from print anyway.

The easiest place to get at the composite video signal is the point where it feeds into the TV modulator. You can find the modulator quite easily. It is the little metal box with the socket where you plug in the TV aerial lead. You will have to open the computer case to get at it.

At one end of the box, there are two wires leading in. The nearest of these to the TV socket is the composite video signal, the other is the modulator's power supply. You need only solder the core of your coaxial lead to the signal wire, and the outer sheath to the metal box itself, for an earth. Fit the other end of the coax with the right kind of plug for your composite monitor, usually a BNC or phono type, and you're away.

Give it a tweak!

Besides dedicated monitors, you can get television sets that have composite video inputs, usually meant for use with video cameras. These will accept the colour video from the Spectrum and provide an improvement over the TV signal because you have cut out the modulator in the computer and the demodulator in the TV set, so the signal is messed about much less. However, bypassing the modulator won't eliminate dot crawl, because that is introduced during the coding into the composite video signal, not during modulation.

The Issue 1 and 2 circuit boards have preset controls for setting up the colour signal circuits. These are adjusted at the factory and you are meant to leave them alone, but it is possible to effect a useful improvement to the Spectrum's display by tweaking them to suit your particular TV set. Indeed, some TVs would not work in colour with these early Spectrums until this was done – and some still wouldn't work after it was done!

First let us eliminate those Spectrums which will not gain anything from this exercise. All Spectrum Pluses and the later rubber keyboard Spectrums with Issue 3 circuit boards do not even have the presets fitted. Instead they have improved ULA circuitry that is self adjusting.

You won't be able to turn a TV into a monitor with a few twists of a screwdriver. It may be best to leave your pots alone if the display is anything like reasonable. After all, you might make it worse. If your whites are yellow or you have trouble getting the colour to lock, then tweaking ought to help. If your only complaint is dot crawl, then you should be able to get an improvement, but it will come and go. You should do these adjustments when the computer is warmed up, as they tend to drift as the temperature changes.

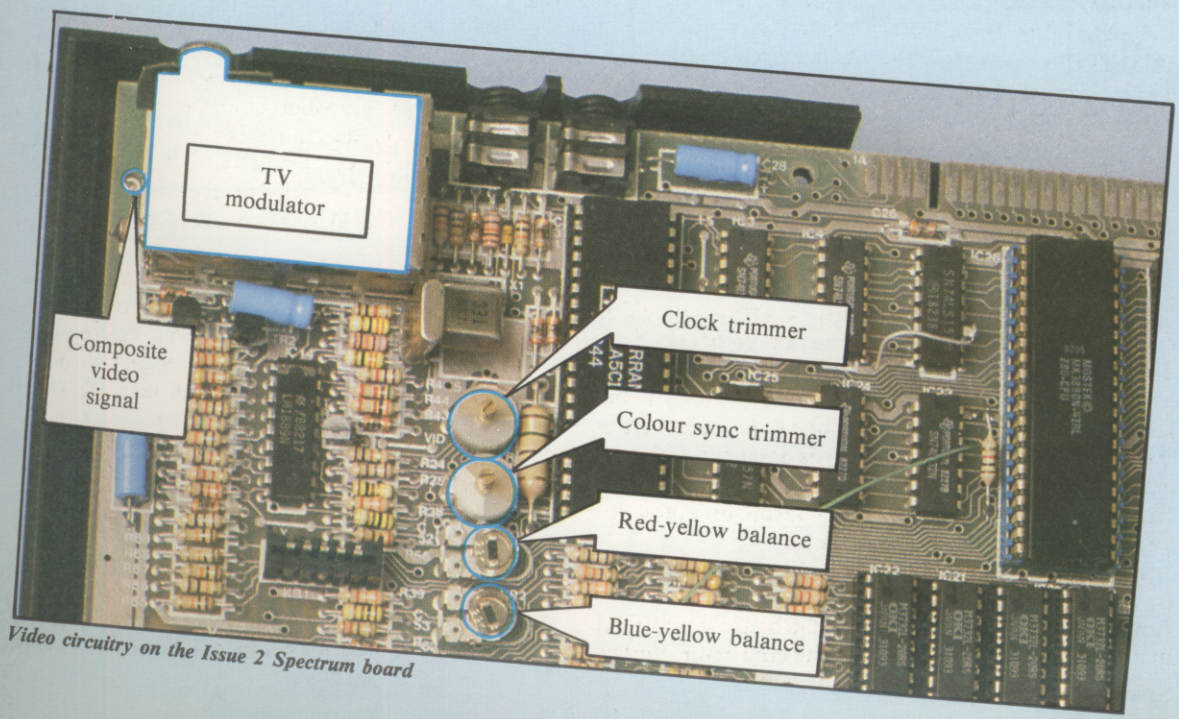
Adjustments must be made with the power on, so that you can gauge the effect on the screen. This is not dangerous to you, as the voltages are all very low, but you must take care not to touch components other than the presets with the end of your screwdriver, as even a momentary short circuit could damage the computer. These presets go through their whole range of adjustment in a single 360 degree turn, so you need only turn the screw a tiny amount to make a difference. Take care not to bend the keyboard ribbon cables about too much, as they are fragile and tear easily.

There are two preset capacitors and two variable resistors all in a row. The two nearest to the TV modulator with the little brass screws are the capacitors. They are used to trim the frequencies of the two crystals that control the computer clock and the colour sync. The one nearest the modulator is the clock trimmer, and this is the one you tweak to minimise dot crawl. The colour sync trimmer should only be adjusted if you have no colour at all, and even then you should first check that the problem is not the tuning of the television set itself.

The two furthest from the modulator are the colour balance controls. These don't have proper screw heads, only an oblong slot, and you may have difficulty finding a screwdriver to fit properly. One controls the balance between red and yellow and the other between blue and yellow and both together effect the green.

These should be tweaked to give good colours, but also to obtain a nice neutral white. Obviously you will want to run a program or give instructions to the computer to get some colours into the display to judge by.

It is possible to get into a complete muddle and mess up the display entirely. Don't panic if this happens, just make a cup of tea and have a good think about what you're doing. There is nothing permanent or irretrievable about any of the adjustments you're making, and eventually it will come right.



Video circuitry on the Issue 2 Spectrum board

INK and PAPER

In the second part of this series, we describe how to display your graphics in glorious colour

SO far the graphics we've been examining have been in black and white. Of course the Spectrum, as should be obvious from its name, is capable of producing coloured text and graphics. This is achieved using the commands INK and PAPER.

First we'll look at INK, which selects the foreground colour – the one you're actually writing in. The actual colour you want to write in is indicated by a number following the INK statement. The colours have the numbers 0 to 9 associated with them, as you'll see from Table I. The following line uses INK to select colour 2, which is red:

```
INK 2
```

The ink numbers, as they're known, are fairly straightforward between 0 and 7 – just the Spectrum's standard colours. We'll ignore ink numbers 8 and 9 for the moment. Once a new INK has been selected it is remembered by the Spectrum and all further symbols produced will appear in the same colour, unless another INK command is executed. It is also possible to change temporarily the INK colour by including an INK command within a statement. For example, the program below prints one line of text in red, one in yellow and another in red:

```
10 INK 2
20 PRINT "HELLO HELLO"
30 PRINT INK 6;"HELLO HELLO"
40 PRINT "HELLO HELLO"
```

Line 30 is the clever bit, because it selects colour 6, yellow, and prints the message "HELLO HELLO" without altering the "general" INK 2 – red – selected by line 10. When INK is included in statements in this way it must be followed by a semi-colon, otherwise the Spectrum gets confused and generates an error.

The second command associated with colour is PAPER. This selects the colour of the background that appears behind the foreground colour. The PAPER colour is specified in exactly the same way as INK. Thus to select a blue background we would use:

```
PAPER 1
```

When a clear screen command, CLS, is executed the screen is cleared with the current background, PAPER, colour. Therefore if the PAPER is changed to 4, green, and CLS is used, the whole screen will become green. This is a very quick way of changing the display to a complete block of colour.

PAPER can also be changed temporarily within a statement in exactly the same way as INK. This is shown in the program:

```
10 INK 7
20 PAPER 1
30 PRINT "WHITE ON BLUE"
40 PRINT PAPER 2;"WHITE ON RED"
50 PRINT "WHITE ON BLUE"
```

As the previous example shows, INK and PAPER can be mixed freely so that different coloured writing can be produced on different backgrounds. The next program shows this further. This time RND is used to select random foreground and background colours.

```
10 INK INT (RND*8)
20 PAPER INT (RND*8)
30 PRINT "This is a message"
40 GO TO 10
```

Outside the screen

The area of the television display around the Spectrum's screen can be changed to different colours with the command BORDER. When you turn the Spectrum on the BORDER and screen background colours are the same. But by entering a BORDER command you can alter the part of the display unused by the screen to any of the eight colours available. As with INK and PAPER, a number follows the instruction to indicate the new colour. For example, the border can be made cyan with the statement:

BORDER 5

0	Black
1	Blue
2	Red
3	Magenta
4	Green
5	Cyan
6	Yellow
7	White
8	Transparent
9	Contrast

Table I: INK colours

Special effects with INK numbers 8 and 9

AS far as INK and PAPER are concerned the numbers 8 and 9 are distinctly odd. They are special in that they don't directly specify a new colour but indicate the way in which the new colour will be found. Let's see what effect they have on INK.

If INK 8 is chosen, the foreground colour that appears depends upon the current foreground colour of the position being printed to. Therefore if a yellow character is overprinted with a letter A, the A will take on the colour already present – yellow in this case.

This has the advantage that, if you're altering something on a screen you've already set up, the alterations don't stand out since they're made in the foreground colour of the original. If you like, it's a sort of transparent ink. Key in this program and see what happens:

```
10 INK 1
20 PRINT "I'M IN BLUE"
30 INK 2
40 PRINT "I'M IN RED"
50 INK 8
60 PRINT AT 0,0;"ALSO IN BLUE"
```

First of all the screen is cleared with CLS and two lines of text are printed at the top of the screen, one in blue the other in red. Next the transparent ink is selected and the cursor is positioned at text location 0,0 – the top left corner. Now another message is printed on top of the previous blue message. Since INK 8 is being used, the blue already there is used for the foreground colour.

The second "odd" colour, 9, selects either black or white INK depending upon the background colour of the position being printed to. If the background is light the INK is changed to black so that the two contrast nicely. If, however, the background is a dark colour, white INK will be chosen instead. This makes it easier to read text which has been printed over an unknown background colour.

This example shows that black text will be printed to contrast with the light coloured yellow background already there:

```
10 PAPER 6
20 INK 2
30 PRINT "RED ON YELLOW"
40 INK 9
50 PRINT AT 0,0;"BLACK TEXT"
```

PAPER 8 works like INK 8, altering the background colour to that of the current print position. PAPER 9, contrast, works in a similar manner to its INK equivalent but uses the current INK (foreground) colour to determine the new background colour.

Charting your graphics progress

TO round this section off, here's a simple program which creates a bar chart type diagram to allow comparisons to be made between data for two years. The data for each year is held at the end of the program. This determines the height of 12 bars for each month in the year. In fact, the bars consist of columns of asterisks.

To distinguish between the two years the asterisks are printed in different INKs. To aid comparisons between the two sets of figures we've done some pretty nifty work with PAPER 8. The data bytes for each column are first read into the single dimension array *a(x)* before being displayed in graphical form.

The program contains two subroutines to save repeating similar lines of instructions. The first is at line 400 and is responsible for creating the columns of asterisks using the data held in the array *a(x)*. The second subroutine – line 500 – reads 12 data bytes into the array *a(x)*. The data is defined in lines 600 to 630.

When the program is RUN the first 12 bytes of data are read by calling the subroutine at line 500 – see line 30. Next the BORDER is changed to white – colour 7 – and the PAPER is changed to yellow – colour 6. The CLS in line 50 then clears the screen using the PAPER colour to result in a solid yellow screen with a white BORDER.

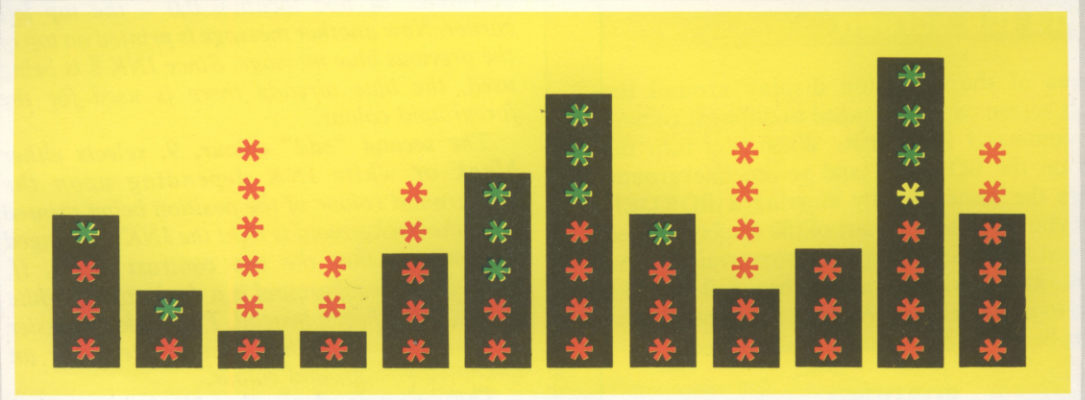
Lines 60 and 70 now change PAPER to black and INK to green. The next line calls the subroutine at line 400 which prints the 12 columns of asterisks on the screen, displaying the information for the first year. The number in the array directly corresponds to the height of the columns. Thus if the first array element contains four the bar will be four characters high.

When the subroutine has finished, the program continues executing from line 90. This changes the PAPER to colour 8 – transparent – and the INK to red – colour 2. Lines 95 and 100 read the final 12 bytes of data and display the corresponding asterisks.

Since PAPER 8 has been selected, all characters printed will take on the background colour already there. So if we try to print one of our red asterisks where there's already a green one from the first year, while our new asterisk will overprint the old it will retain the black background of the old data. So if we see a red asterisk on a black background we know that "height" had been reached the previous year.

If, however, we're printing a red asterisk above the level reached the previous year, it will adopt the yellow background already there.

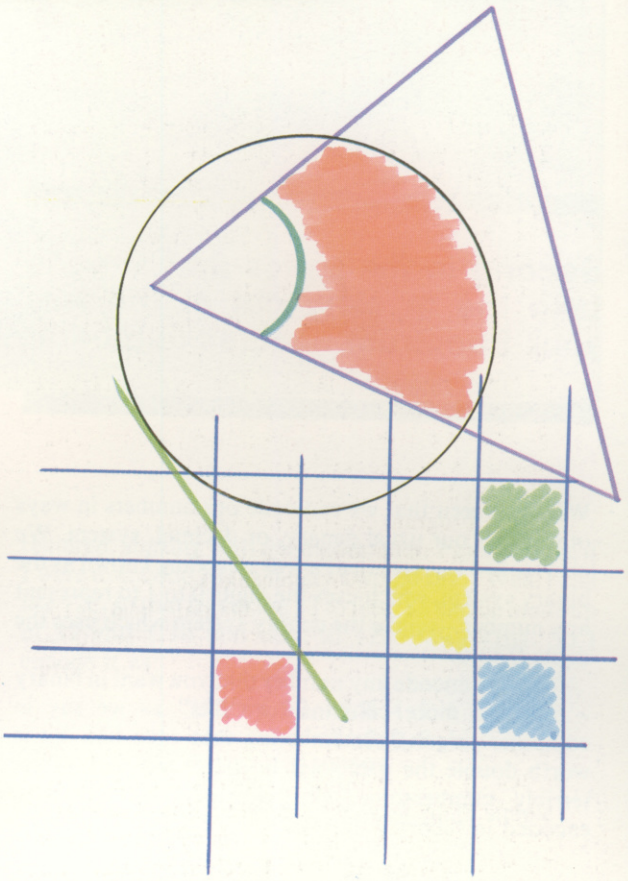
All this means that it is now possible to see the differences in the two sets of data by comparing the height of the black background given by the first year's data, and the red asterisks from the second. This would not have been possible without the useful transparent colour option.



```
10 REM bar chart
20 DIM a(12)
30 GO SUB 500
40 BORDER 7: PAPER 6
50 CLS
60 PAPER 0
70 INK 4
80 GO SUB 400
90 PAPER 8: INK 2
95 GO SUB 500
100 GO SUB 400
110 STOP
400 FOR n=1 TO 12
410 FOR m=1 TO a(n)
420 PRINT AT 21-m,n*2;"*"
430 NEXT m
440 NEXT n
450 RETURN
500 FOR n=1 TO 12
510 READ a(n)
520 NEXT n
530 RETURN
600 DATA 4,2,1,1,3,5
610 DATA 7,4,2,3,8,4
620 DATA 3,1,6,3,5,2
630 DATA 4,3,6,3,5,6
```

Graphic details

All the facts you need to know about those graphics commands – at your fingertips



PLOT x,y — Q key

Places an INK spot at pixel coordinates x,y, subject to INVERSE and OVER. Moves the start position for DRAW to x,y. The pixel coordinates are:
x= across the screen left to right, from 0 to 225
y= up the screen from the bottom, from 0 to 175.

DRAW x,y — W key

Draws a line from current PLOT position to a point displaced from it by x pixels horizontally and y pixels vertically. (Use minus numbers to move left or down.)

DRAW x,y,z — W key

As above, but draws a curve between the two points of an angle z, where z is expressed in radians. (Experiment!)

CIRCLE x,y,z — H key, Ext. mode

Draws a circle with a radius of z pixels with the centre at x,y.

System Variables

23606: Two byte address which is 256 less than the address of the bit patterns of the character set. The normal set is in the ROM, but you can define your own set and POKE this address to point to it. The character set bit pattern is defined in the same way as the UDGs.
23675: Two bytes, the address of the first user defined graphic. You can have more than one set of UDGs, and POKE the address to switch between the different sets.
23677: The x coordinate of last pixel plotted.
23678: The y coordinate of the last pixel plotted.

POINT (x,y) — 8 key, Ext. mode

Returns a 1 if the pixel at x,y is INK, or a 0 if it is PAPER.

USR "a" — L key, Ext. mode

Returns the start address for the bit pattern of UDG "a" (here a stands for any letter between a and u). This address and next seven addresses may be poked to define the UDG – this is usually done with a program loop reading and poking from DATA statements. At a power-up the UDGs are predefined to look like capital letters. Each one is defined by a 8 × 8 matrix using eight bytes of binary. There are 21 UDGs, and 21 × 8 = 168 addresses to POKE to define all of them. This area is usually at 65368 in a 64k Spectrum, but can be moved (see adjoining panel).

Graphics Mode — 9 key + Caps Shift

When in this mode, press letters from "a" to "u" to get User Defined Graphics – these look like letters until redefined – or press numbers 1 to 8 to get block graphics (with Caps Shift for inverted form). Cursor becomes G in graphics mode. Press 9 key again to leave graphics mode.

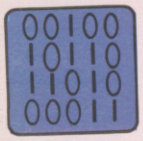
Take a byte...

Second in the series that aims to take the mystery out of the way your Spectrum handles numbers

WE have seen that we can code our numbers in ways other than our usual denary, or decimal, system. We also looked last month at a way of coding known as the binary system, which uses the digits 0 to 1 to represent any number – unlike the denary system which uses the digits 0 to 9.

You can encode any number that you want in binary – just use more columns (or "bits" as we say in computer jargon), remembering that each new bit is worth double the preceding bit. However it does get terribly cumbersome. For example, 100 (denary) encoded in binary is %1100100 since:

```
    64 32 16  8  4  2  1
%   1  1  0  0  1  0  0
→   64+32+4=100
```



It is much easier to handle the number in our normal system. To a computer this presents no problem, and the fact that binary only uses two symbols is a bonus because you can represent numbers with a sequence of "switches".

State of the art

SWITCHES are what we call "two state" – they're either ON or OFF. If we have a sequence of four switches together we can encode numbers by having them either ON or OFF. We could use ON to mean a 1, and OFF to mean a 0 in a particular column:

```
     8    4    2    1
    ON   OFF  ON   ON
→ %  1    0    1    1 = 11
```

Each of these "switches" represents a bit, and a computer memory is full of bits. The Z80, which is the microprocessor at the heart of the Spectrum, deals with 524,288 of them. To make things simpler the Z80 handles the bits in groups of eight bits at a time – the group of eight being called a byte. With this type of organisation the largest number you can store in a byte is 255 since:

```
    128 64 32 16 8 4 2 1
%   1   1  1  1  1 1 1 1
→ 128+64+32+16+8+4+2+1=255
```

Of course the computer can handle larger numbers *(and not just whole numbers)* but to do so it must use more than one byte. Converting a byte from binary to denary is fairly straightforward. Simply write it down under the appropriate column (or bit) values and add together the value of all the columns in which a 1 occurs. For example, given %10010101 you translate as follows:

```
    128 64 32 16 8 4 2 1
%   1   0  0  1  0 1 0 1
→ 128+16+4+1=149
```

Going from denary to binary is not at all difficult, but is rather hard to put into words. You do it by subtracting from the number you want to encode the value of each column in turn, starting with the highest (that is 128, 64, 32 and so on).

If you can subtract a particular column value you put a 1 in that column and continue to subtract the next lower column value from the remainder. If you cannot manage the subtraction you put a 0 in that column and try to repeat the subtraction with the next lower column number. So, starting with the highest column number (128 in our case), you:

1. Attempt to subtract the relevant column number

Doing your sums the binary way

JUST as we can do sums with decimal numbers, so we can with binary numbers, though they'll only involve 0s and 1s. Take a look at the following:

```
  % 1
+ % 1
  %10
```

If you think about it, that is correct, since the sum adds one and one, and the answer %10 is binary for two. One way of relating this to our usual way of doing sums is to say that we carry when we get to two, instead of ten as we do in our normal, decimal, sums.

Another way to look at it is that we have to carry when we get to two because we aren't allowed to use the digit '2'. If you remember, in part one we had a rule forbiding two "coins" of the same value. Try this sum:

```
    4 2 1
  %   1 1  in        3
+ %   1 0  denary  + 2
  %1  0 1            5
```

Here we carry from the second column to the third. Addition is not very hard at all – just make sure that you always "put 0 down and carry 1" when you get a two. If you get a three then "carry

149	
−128	128 goes – set to 1
21	64,32 can't go – set to 0
−16	16 goes – set it to 1
5	8 can't go – set to 0
4	4 goes – set to 1
1	2 can't go – set to 0
−1	1 goes – set to 1
0	

128	64	32	16	8	4	2	1
1							
	0	0					
			1				
				0			
					1		
						0	
							1
% 1	0	0	1	0	1	0	1

Figure I: Converting from denary to binary

(highest first).

2. If you succeed then put a 1 in that column number and continue to subtract other columns from the remainder. If not put a 0 in that column. If you have not done all eight columns repeat from step 1.

Figure I should make it clearer.

In practice, when faced with encoding a number from denary to binary I tend to do it in my head, seeing which column values will add together to make the sum required, starting with the highest first. For example, if I were to encode 161 in binary I would say, "Well, I can use 128, so that leaves me 33 to find. 33 can be made up of 32 and 1 so that does it: 128+32+1=161. So I encode it as:

```
      128 64 32 16 8 4 2 1
%       1  0  1  0 0 0 0 1
      = %10100001
```

After a while you'll find this way quite simple.

one for two and put one down". For example:

```
      8 4 2 1
%     1 1 1 in        7
+ %     1 1 decimal  +3
  % 1 0 1 0          10
```

Subtraction is a little more complicated, and depends on whether you borrow or decompose! The latter phrase doesn't describe the current economic climate, it's just that there are two schools of thought on the way subtraction should be taught – the borrowers and the decomposers.

Fortunately, we can ignore binary subtraction since we can manage without it – as does the microprocessor inside your machine. If you want to do some binary subtraction it is straightforward enough provided that you remember that it is two you're borrowing or taking, not ten. Figure II illustrates the process – without any attempt to explain it.

Before we leave the realm of simple sums, look what happens if we shift everything in a binary number over to the left, putting a zero into bit 0, which would be left vacant otherwise. For example:

```
      8 4 2 1
%     1 0 1     which is 5
```

becomes

```
      8 4 2 1
% 1 0 1 0       which is 10
```

This shifting to the left doubles the number automatically. This isn't too hard to visualise, because the value of each bit is transferred to the next higher bit, which is of course double in value – so the end result is that the whole number is doubled in value. Similarly, we can do the binary equivalent of dividing by 2 by shifting to the right. For example:

```
      8 4 2 1
%     1 1 0 1     which is 13
```

becomes

```
      8 4 2 1
%       1 1 0     which is 6
```

Of course, 13 divided by 2 should gives you 6 remainder 1, but this technique ignores remainders. This sort of division, where we're only interested in the "wholes" is known as integer division. It's equivalent to INT(13/2).

As each bit is moved to the right, it occupies a column exactly one half lower in value, thus the sum total of all the bits is one half lower, save for the original bit 0 which has disappeared altogether (hence the ignored remainder).

```
% 1 1 0        % 1 1 1
−%   1 1   OR  −%   1 1   In decimal  6
% 1 1 1        % 1 1 1                −3
                                       3
Decomposition   Borrowing
```

Figure II: Binary subtraction

More power to your number

AS we've said, a byte contains eight bits. These bits are labelled, somewhat peversely, bits 0 to 7, as you'll see from Figure III. Bit 0, as you can see, is the "1" column. As this is the smallest value bit we say that bit 0 is the least significant bit (LSB). Bit 7, the "128" column, is called the most significant bit (MSB). The reason for using the numbers 0 to 7 to label the bits instead of the more logical 1 to 8 has to do with powers, a subject you almost certainly covered at school.

"2 to the power 2" is 2*2 = 4
"2 to the power 3" is 2*2*2 = 8
"2 to the power 4" is 2*2*2*2 = 16

and so on. "2 to the power 8" would be eight twos all multiplied together. Notice as the powers of two increase – that is, as we multiply more twos together – the answers are doubling, just as our column or bit values do.

Also, 2 to the power of 2 is 4, the value of bit 2, while 2 to the power of 3 is 8, the value of bit 3. It shouldn't come as any surprise to you to find that 2 to the power of 7 is 128, the value of bit 7.

You can verify this on the Spectrum by using the symbol "↑" which stands for "to the power of". It shares a key with the letter H.

Try:

```
PRINT 2↑4
PRINT 2↑7
```

Be sure to try 2↑1, which will show you why bit 1 has the value 2. Also try 2 0. The answer may surprise you. The fact is that any number to the power 0 is 1! Hence bit zero has the column value of one. Figure IV illustrates this.

From decimal to binary

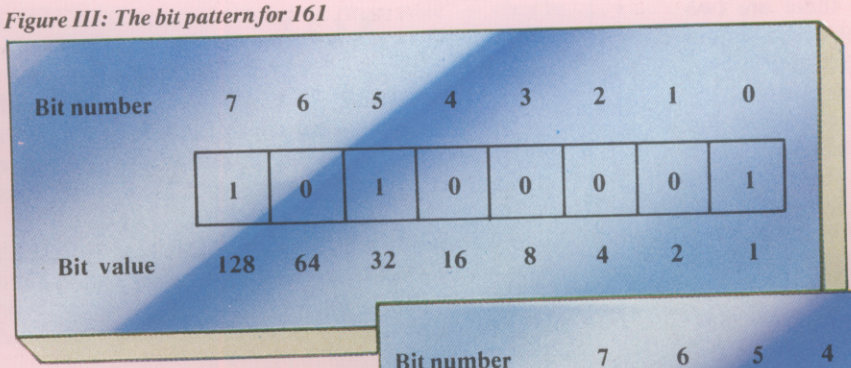
AS we've seen, the Spectrum has a simple way to convert binary numbers to denary – the BIN function. However there's no simple Basic instruction to allow you to go the other way, from decimal to binary.

Well, here's a simple program to do the work for you. It works by repeatedly taking away powers of two from the target number:

- *If you can take a particular power of two away from the number, the column corresponding to that power has a one in it (line 110).*
- *If, on the other hand, it won't go, the relevant column has a zero put in it (line 120).*

```
  10 REM decimal to binary
  20 INPUT "Enter a decimal number",n
umber
  30 IF number>=0 AND number<256 THEN
  GO TO 80
  40 PRINT "Your number must be betwe
en 0"
  50 PRINT "and 255!"
  60 FOR l=1 TO 500: NEXT l
  70 PRINT : GO TO 20
  80 PRINT number;" in binary is:"
  90 PRINT
 100 FOR l=7 TO 0 STEP -1
 110 IF (number-2^l)>=0 THEN  LET num
ber=number-2^l: PRINT "1";: GO TO 130
 120 PRINT "0";
 130 NEXT l
 140 PRINT : PRINT
 150 GO TO 20
```

Figure III: The bit pattern for 161



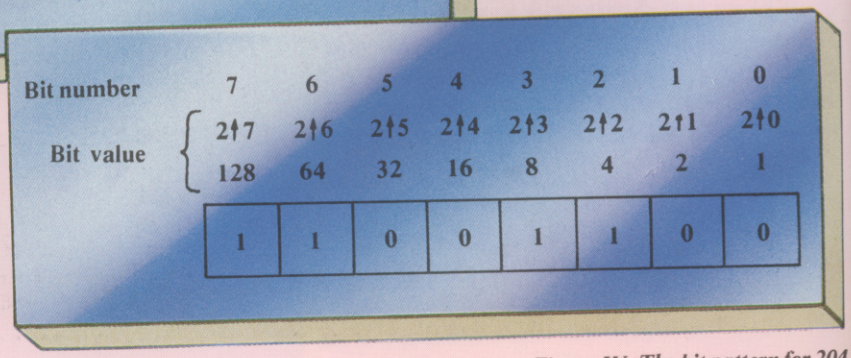


Figure IV: The bit pattern for 204

We thought it was about time we put you in the picture.

When we introduced our AMX Mouse to micro-users, the response was phenomenal.

And no wonder!

Hailed by the press as 'probably the best input device that has arrived recently', the AMX Mouse brings to Spectrum 48K users, the same sophisticated, positive control that has, until now, been the province of more expensive computers – like the Macintosh.

The fantastic AMX Mouse Package opens up an entirely new and exciting world to Spectrum users and comes complete with Mouse, interface, which also includes a Centronics printer interface, and all these fabulous programs.

AMX ART

This computer aided, drawing program has to be seen to be believed. Making full use of on-screen windows, icons, pull-down menus and pointers, you'll be astonished at the quality of the work you can produce, save and print using either ZX or Epson compatible printers. It's a program ideal for both hours of family fun or for serious professional applications.

AMX COLOUR PALETTE

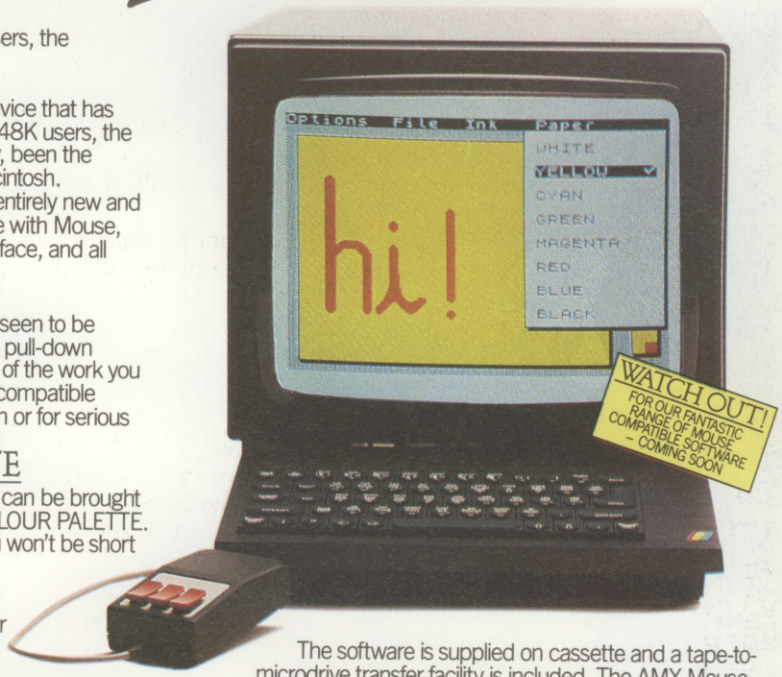
The wonderful pictures you create with AMX ART can be brought vividly to life with rich vibrant colours – using AMX COLOUR PALETTE. And with 8 Foreground and 8 Background colours you won't be short of inspiration.

AMX CONTROL

Now you can create a 'Mouse environment' in your own programs, AMX Control adds 28 commands to normal Sinclair Basic and contains three programs.

1. The machine code program to extend the Basic interpreter, which gives you full use of windows, icons, pull down menus, pointers and also supports AMX printer interface.
2. An Icon designer, an individual program for creating and storing icons for use in your own programs. The number and variety of icons you can create is limited only by your needs and imagination.
3. A demonstration program containing on screen calculator and puzzle.

This fabulous AMX Mouse Package costs only £69.95, a price as remarkable as the package itself and it includes a fully illustrated operating manual.



The software is supplied on cassette and a tape-to-microdrive transfer facility is included. The AMX Mouse package is compatible with the Spectrum 48K and Spectrum +.

Ordering couldn't be easier.

This superb package is available from all good computer dealers or direct using the freepost order form below.

So what are you waiting for?

Get into the picture now!

☎ FOR INSTANT ACCESS OR VISA ORDERS RING (0925) 602959/602690

PLEASE RUSH ME POST FREE_____(Quantity)
AMX MOUSE SPECTRUM 48K PACKAGE/S
AT £69.95 EACH INCL. VAT AND P & P.

I ENCLOSE CHEQUE/POSTAL ORDER FOR £_____

OR DEBIT MY ☐ ACCESS
(TICK AS APPROPRIATE) ☐ VISA

CARD NO. ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐

EXPIRY DATE_____

SIGNATURE _____

NAME _____
(BLOCK CAPITALS PLEASE)

ADDRESS _____

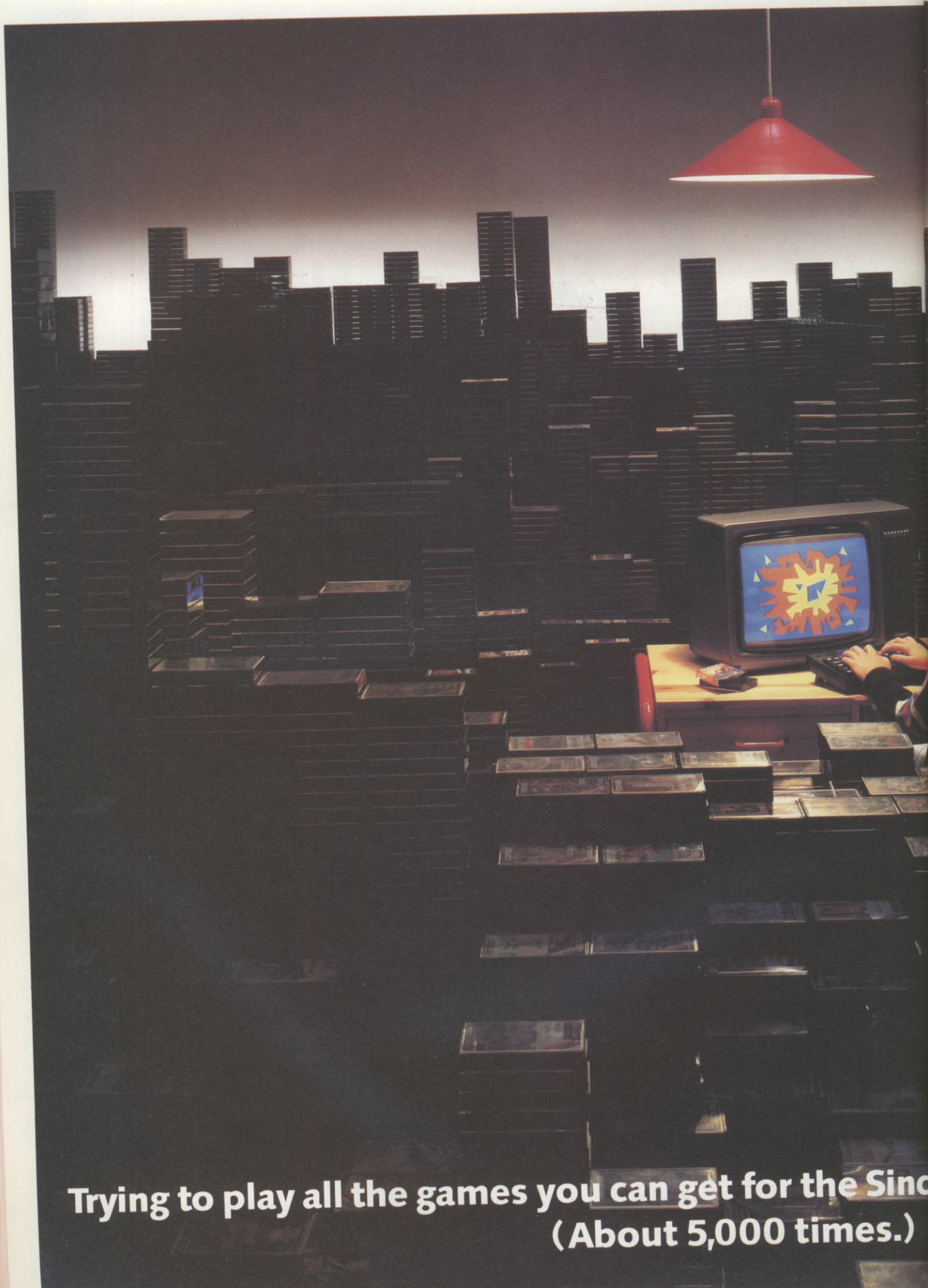
_____

_____ POSTCODE _____

AMX MOUSE

IMAGINATION AT YOUR FINGERTIPS

SEND TO: ADVANCED MEMORY SYSTEMS LIMITED, FREEPOST, WARRINGTON WA4 1BR.


Trying to play all the games you can get for the Sinc
(About 5,000 times.)

nclair Spectrum could kill you.

.)

How would you like to be blasted out of the sky by anti-aircraft guns?

Or perhaps you'd prefer to be vapourised by a Thargoid starship.

Or eaten for breakfast by mutant rats.

Well, you have all this and more to look forward to, if you get a Sinclair Spectrum +.

Because there are more games available for the Spectrum + than any other home computer.

(About 5,000 the last time we counted.)

Not all of which lead to such sticky ends, we might add.

You may, for instance, rather battle it out on the chess board with a Russian grand master. Or cross swords with Nick Faldo on the golf course.

You won't just have the pick of the current titles either.

You'll get first shot at the new ones too. Because most of them become available for the Spectrum + before anything else.

And because there are so many other Spectrum owners there are more books and magazines and clubs to join.

So if you get a Spectrum +, you may well end up in a Grand Prix pile-up.

And your chances of being killed by Gremlins are very high.

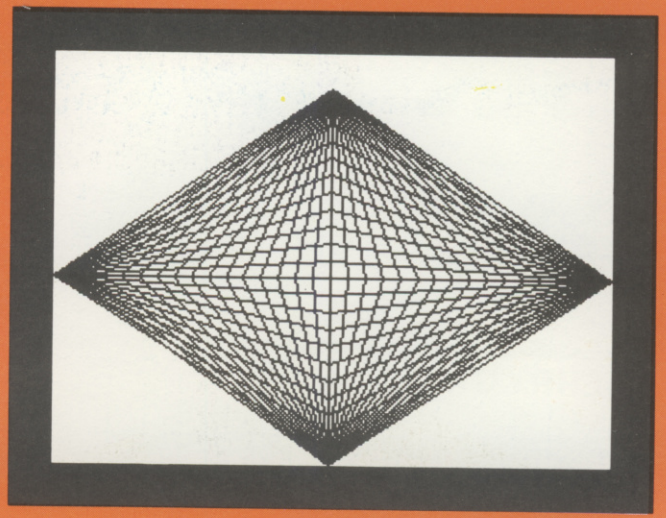
But you'll never die of boredom.

sinclair

BRABHAM STRAIGHT
GRAHAM HILL BEND
PADDOCK HILL
HAILWOOD RISE
DRUIDS
KEMPSTON
MICRO ELECTRONICS LTD
NEW GRAND PRIX SERIES!
FORMULA
A.R.C.
COMPETITOR/
MECHANIC
APR 1985
BRANDS
Available directly from Kempston Micro Sales Ltd Fast Mail order service. Please send me the following
Name
Address
Postcode
1 Kempston Joystick Interface £9.95
2 Centronics Interface E £39.95
3 Formula 2 Joystick £11.95
4 Kempston Pro Joystick Interface £16.95
5 Formula 1 Joystick £16.95
6 Junior Pro-Joystick £5.95
SPECIAL COMBINATION OFFER
Formula One Joystick + Joystick Interface £19.95
Formula Two Joystick + Joystick Interface £17.50
Access/B'card
All Prices include VAT P&P. Overseas add £4 P&P
Kempston Micro Sales Ltd, Singer Way, Kempston, Bedford MK42 7AW.
Tel (0234) 856633 Telex 826078 KEMPMI G
NEW GRAND PRIX SERIES!
DANGER: WARNING: THESE PRODUCTS WILL SERIOUSLY DAMAGE OUR COMPETITORS MARKET.
JOYSTICK

Analysis

In this series we present a detailed breakdown of a short but interesting program. The first listing shows how PLOT, DRAW and GOSUB can combine to form striking graphic effects



```
10 REM Network
20 LET drawline1=8000
30 LET drawline2=9000
40 CLS
50 PLOT 128,0:DRAW 0,175
60 PLOT 0,88:DRAW 255,0
70 FOR a=1 TO 2
80 GO SUB drawline1
90 NEXT a
100 FOR b=1 TO 2
110 GO SUB drawline2
120 NEXT b
130 GOTO 130
8000 FOR l=0 TO 255 STEP 8
8010 PLOT l,88
8020 IF a=1 THEN DRAW 128-l,87
8030 IF a=2 THEN DRAW 128-l,-88
8040 NEXT l
8050 RETURN
9000 FOR m=0 TO 175 STEP 8
9010 PLOT 128,m
9020 IF b=1 THEN DRAW -128,88-m
9030 IF b=2 THEN DRAW 127,88-m
9040 NEXT M
9050 RETURN
```

This REM is ignored by the Spectrum but acts as a program label for humans. The program will work without it.

Give values to the numeric variables *drawline1* and *drawline2*. These contain the line numbers of the two subroutines that are used.

The PLOT puts a dot at the bottom of the screen, in the middle. The DRAW draws a line from this to the top of the screen.

Clears the screen.

Draws a line across the middle of the screen.

Each time the loop is performed, this line calls the subroutine at the line number stored in *drawline1*. Notice how using a variable name tells us what the subroutine is to do. The program is more readable.

Form a FOR ... NEXT loop which goes round twice. The first time round the loop control variable has the value 1, the second time its value is 2.

Make up another FOR ... NEXT loop control variable *b*. This cycles twice as *b* takes the values 1 and 2.

Invokes the second subroutine.

Sets the start of each line to new point on the line going across the centre of the screen.

This endless loop stops the program "crashing" into the following subroutine definitions. It also keeps the screen tidy. Leave it out and see what happens next.

Depending on the value of *a*, one of these lines is performed. They draw from the previously plotted point to the middle of either the top or the bottom of the screen.

Form the first subroutine. It draws the slanting lines from the midpoints at the top and bottom of the screen.

Make up yet another FOR ... NEXT loop. As *l* takes values from 0 to 255, in steps of 8 at a time, so the lines move across the screen making a fan shape.

Ends the subroutine and sends the program back to the statement after the one that originally called it.

Each time round the loop this picks a point further up the centre vertical line.

Make up the second subroutine, drawing the slanting lines from the left and right of the screen.

The first time the subroutine is called this line draws the lines to the middle left of the screen. The second time *b* is not equal to 1 so the line is ignored.

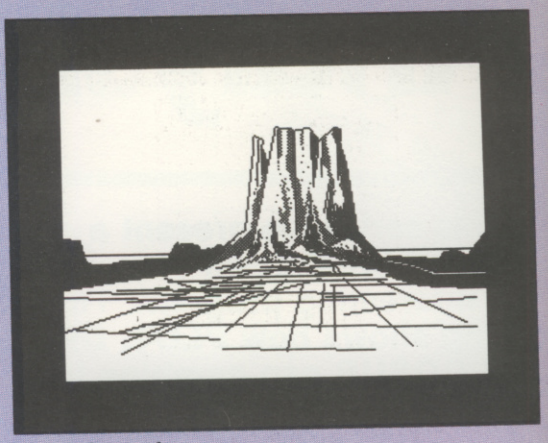
The range of the FOR ... NEXT loop is smaller in this subroutine because the screen is smaller from top to bottom than from side to side.

Sketches in the lines from the right of the screen when *b* is equal to 2, the second time the subroutine is called.

Once you've understood how Network works see if you can make it more efficient. Is there any need for two subroutines? Would just one do the job? And when you've answered that, use it as a basis for your own pattern-producing programs.

Get things moving!

TERRY GREER begins his easy-to-follow series on animation techniques



The background . . .

ANIMATION is basically an illusion – nothing actually moves, but when a series of pictures, each slightly different, is shown in rapid succession the eye and the brain interpret this as movement. This phenomenon is called persistence of vision.

Of course it isn't quite that simple. For the brain to be fooled properly you have to meet several other conditions. These are:

- The difference between pictures, or frames, as I shall refer to them from now on, must be small.
- The time interval between frames must be small.
- The time taken to draw a picture must be much shorter than the interval between frames.
- Any apparent difference between two consecutive frames should be regular. For example, if a ball moves three pixels between frames 1 and 2 it should move three pixels between frames 2 and 3. (A pixel is the smallest dot of colour the Spectrum can put on your screen.)

There are of course always exceptions. As with every conjuring trick, the timing is critical. Here timing refers to the speed at which one frame is replaced with another. The following examples show the normal number of frames per second shown in different media.

MEDIA.	Frames per second
Prof. Cinema 35mm	24
Television	25
Amateur Film 16mm	24
Amateur Film 8mm	16-18

Working flat out, the Spectrum couldn't come anywhere near to placing such number of screens per second on to your TV and would run out of memory after about a third of a second.

Fortunately there are ways around this. First you can get away with about 10 frames a second. Second, and more important, you need only animate a small section of the screen. Third, the same piece of animation can often be used over and over again.

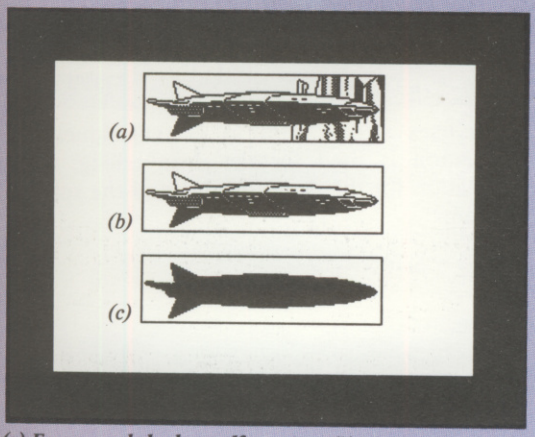
To see these ideas in action, try out Listings I and II which demonstrate the two main types of animation. Listing III combines these techniques to give very sophisticated results. These last two examples make use of the speed at which the Spectrum can print strings of characters and change the system variable CHARS. In the next instalment I'll detail the construction of a walking character and give a listing enabling you to test your own animated graphics.

Listing I: Moving an object about the screen

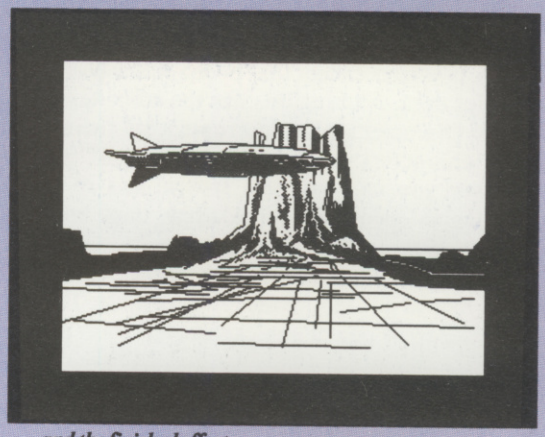
This is a very simple example of moving an object under the control of the cursor keys. Once the program is keyed in and run, pressing keys 5 to 8 – the ones with the arrows on – will move a solid block around the screen. Experiment by altering the graphic character used. Can you make it move more than one "step" at a time?

The animation is crude and jerky, mainly because any movement is always eight pixels – one character block – in any direction and there is no overlap of any two succeeding images. Line 200 checks for movement, only printing if it has

```
  5 REM LISTING 1
  6 REM
  7 REM
 10 LET a=8: LET b=8
 20 LET a1=8: LET b1=8
100 PRINT AT b,a;CHR$ 143
120 IF INKEY$="5" THEN  LET a=a-1
130 IF INKEY$="8" THEN  LET a=a+1
140 IF INKEY$="6" THEN  LET b=b+1
150 IF INKEY$="7" THEN  LET b=b-1
160 IF a>31 THEN  LET a=31: LET a1=a
170 IF a<0 THEN  LET a=0: LET a1=a
180 IF b<0 THEN  LET b=0: LET b1=b
190 IF b>21 THEN  LET b=21: LET b1=b
200 IF b1<>b OR a1<>a THEN  PRINT AT
b1,a1;" ": LET a1=a: LET b1=b
210 GO TO 100
```



(a) Frame needed when self-matting. (b) The object. (c) Matte of the object.



. . . and the finished effect.

Methods of printing graphics

THERE are three main techniques for printing graphics:

SELF-MATTING: This describes what happens normally on the screen when one letter is overprinted by another.

Unless the OVER option on the Spectrum is set to one the new letter or character will overwrite the old one so that only the new one is visible.

This has the disadvantage that it can only be used if there is no background to corrupt or the background is taken into consideration when designing the frames – otherwise holes start appearing.

OVER 1: It is very simple to restore the background to its original appearance simply by printing the same character printed initially, in exactly the same place. Many games use this method.

Its disadvantages are that things can look very strange if they pass over other objects, and that printed objects tend to need to be solid with no hollow spaces (as in Jet Set Willy) otherwise holes in the character will show the background.

occurred. If you omit the check, by deleting from it the following:

IF bl<>b OR al<>a THEN

you will notice that the flickering is present even when the block is apparently stationary. Despite this the illusion of movement is definitely there.

Program structure

Lines	
100	Prints the block (CHR$ 143) at a particular location.
160 to 190	Check that the block never gives an error report by trying to leave the screen.
200	Overprints the object with a blank space, but only if there has been movement of the print position.

THE MATTED OBJECT: This type of printing is becoming increasingly popular with games programmers, as can be seen in Highway Encounter, Knight Lore, Marsport and so on. It is by far the most complex of the techniques described and requires a lot of memory and machine code to develop.

The main background is held in memory untouched as are all the objects. The frame itself and a corresponding matte exist for each object – the matte being the exact shape of the object if it were in silhouette.

The procedure involves:

1. Transferring a blank background to the screen memory.
2. Adding the first object's matte to the background in such a way as to leave a hole the exact shape of the matte (normally using AND/OR in machine code).
3. Adding the detail by printing the object to the screen over the matted background.

Any other objects to be included are added by repeating steps 2 to 3.

Objects are added from those further away (least priority) to those closest (highest priority) in order to give a sense of perspective.

The disadvantages of this method are that it not only requires a large amount of extra RAM set aside as a screen buffer, but also requires twice as much RAM overhead for the actual graphics since a matte is required for every object.

UDGs

IN the Spectrum normally only 21 characters are available to be redesigned as User Defined Graphics (UDGs). All the patterns for the normal character set from CHR$ 32 to CHR$ 127 and for the initial values of the 21 UDGs are held in the Spectrum's ROM and are therefore fixed. The computer's pointers however, which tell it where to look for a particular pattern, are held in RAM and are thus changeable.

One such pointer is the system variable CHARS at addresses 23606 and 23607. Normally 23606 contains 0 and 23607 contains 60. This points to 256 less than the character set in ROM. (The character set starts with a space and continues to the copyright symbol.) Altering this pointer therefore gives you 96 new printable characters – replacing the normal alphabet – with only two pokes. A very powerful function indeed.

You can experiment by poking various numbers into the two addresses, but beware for as soon as you start the character set will become corrupted and illegible. It is therefore a good idea to add a line similar to line 120

in Listing II to restore the normal character set.

System variable UDG, at addresses 23675 and 23676, is similar except that points to the User Defined graphics area. Altering this has the advantage that the normal character set isn't corrupted, but there is a limit of 21 characters that can be "redefined" at a time. However you can maintain several sets of alternate graphics in memory at the same time and switch between them.

When poking these addresses remember it's the low byte which is poked into the lower address. To find out the values to be poked you can use this short program:

```
10 PRINT "INPUT ADDRESS TO BE CONVERTED"
20 INPUT A
30 IF A > 65535 OR A < THEN PRINT "MISTAKE
MADE":PRINT "PRESS ANY KEY TO CONTINUE":PAUSE
0:CLS:GOTO 10
40 LET HI=INT(A/256)
50 LET LO=A-(HI*256)
60 PRINT:PRINT "LOW POKE IS ";LO
70 PRINT "HI POKE IS ";HI
80 PRINT:PRINT:PRINT "NEXT?"
90 PAUSE 0:IF INKEY$="Y" OR INKEY$="y" THEN
CLS:GOTO 10
100 STOP
```

And don't forget – if you're using the system variable CHARS, subtract 256 from the address of the new character set before you input it.

Listing II: Static animation

This short program demonstrates what is in my opinion one of the simplest ways of obtaining unlimited graphics, the judicious use of the system variables CHARS and UDG. Again, nothing is actually moving (remember this as the single most important fact).

You will notice that the animation appears very smooth. This is mainly because the successive printings are self-matting, and the apparent difference between consecutive frames is only about one pixel. The Spectrum's speed at printing strings also helps a great deal.

```
 85 REM LISTING 2
 86 REM
 87 REM
 90 FOR n=1 TO 255
100 POKE 23606,n
110 POKE 23607,60
120 IF INKEY$<>"" THEN  POKE 23606,0
: POKE 23607,60: STOP
130 PRINT AT 0,0;"ABCDEFGHIJKLMNOPQR
STUVWXYZ"
150 PRINT "B"
160 PRINT "C"
170 PRINT "D"
180 PRINT "E"
190 PRINT "F"
200 NEXT n
300 GO TO 90
```

Listing III:

The dripping tap

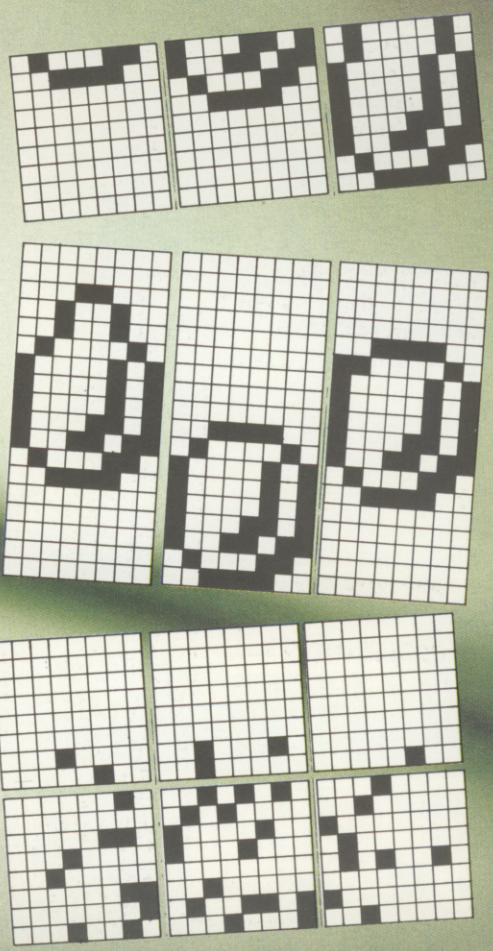
This uses a combination of techniques. Moving animation is used for the drop falling, and static animation for the drop forming and the splash at the bottom.

```
  2 REM LISTING 3
  3 REM DRIPPING TAP
 10 CLEAR 39999
 20 FOR n=1 TO 120: READ a: POKE 399
99+n,a: NEXT n
 30 LET a$="": FOR n=32 TO 52: LET a
$=a$+CHR$ n: NEXT n
 35 GO SUB 2000
 40 POKE 23606,64
 50 POKE 23607,155
 60 LET p=5
 90 REM DROP FORMING
 91 REM
 92 REM
100 PRINT AT 5,10;A$(1)
105 PAUSE p
110 PRINT AT 5,10;A$(2)
115 PAUSE p
120 PRINT AT 5,10;A$(3)
125 PAUSE p
130 PRINT AT 5,10;A$(4)
135 PRINT AT 6,10;A$(5)
140 REM DROP FALLING
141 REM
142 REM
150 FOR n=1 TO 14
160 PRINT AT 4+n,10;A$(6)
165 PRINT AT 5+n,10;A$(7)
170 PAUSE 1
180 PRINT AT 5+n,10;A$(8)
185 PRINT AT 6+n,10;A$(9)
190 NEXT n
195 REM SPLASH
196 REM
197 REM
200 PRINT AT 4+n,9;A$(10 TO 12)
210 PRINT AT 5+n,9;A$(13 TO 15)
215 POKE 23606,0: POKE 23607,60
```

Program structure

Lines	
10	Sets RAM TOP to 39999.
20	Pokes into place the data for the graphic characters.
30	Sets up A$ which contains all the characters needed for the program.
40 and 50	The pokes to change CHARS so that it points to the correct area of RAM.
3000 to 3014	Hold data for the 15 graphics characters needed
4000	Redefines the normal character set if a key is held down – this is how you exit the program.

```
 220 PRINT AT 4+n,9;"   "
 230 PRINT AT 5+n,9;"   "
1000 IF INKEY$="" THEN  GO TO 40
1010 GO TO 4000
1990 REM DRAW TAP AND FLOOR
1991 REM
1992 REM
2000 PLOT 0,7: DRAW 180,0
2005 PLOT 128,151
2010 DRAW -15,0: DRAW 0,5: DRAW -5,0:
 DRAW 0,10: DRAW 15,0: DRAW 0,4: DRAW
 -35,0: DRAW 0,-4: DRAW 15,0: DRAW 0,
-10
2020 DRAW -5,0: DRAW 0,-5: DRAW -20,0
: DRAW 0,-15: DRAW 10,0: DRAW 0,5: DR
AW 40,0
2030 RETURN
2990 REM GRAPHICS DATA
2991 REM
2992 REM
3000 DATA 70,60,0,0,0,0,0,0
3001 DATA 141,155,70,60,0,0,0,0
3002 DATA 66,66,133,133,141,155,70,60
3003 DATA 0,0,24,36,36,66,133,133
3004 DATA 141,155,70,60,0,0,0,0
3005 DATA 0,0,0,0,0,0,0,0
3006 DATA 60,66,133,133,141,155,70,60
3007 DATA 0,0,0,0,60,66,133,133
3008 DATA 141,155,70,60,0,0,0,0
3009 DATA 0,0,0,0,0,0,16,4
3010 DATA 0,0,0,0,0,0,34,32
3011 DATA 0,0,0,0,0,0,0,4
3012 DATA 2,0,6,16,32,3,1,18
3013 DATA 42,80,132,136,0,32,77,17
3014 DATA 16,32,128,64,82,0,120,32
4000 POKE 23606,0: POKE 23607,60: STO
P
```



Nybbles & bytes

Making sense of machine code – Part II

IN Part I we had a look at binary numbers, the sort your Spectrum uses. As we saw, a binary number consists of a lot of 0 s and 1 s together, each 1 or 0 being known as a bit. These bits normally come in groups of eight at a time, called bytes.

%10101100 is a typical binary number. Notice the % sign at the front – to distinguish binary from our normal decimal numbers. After all, we don't want anyone mistaking it for ten million, one hundred and one thousand, one hundred, do we?

But how do we interpret it? Well, the values meant by the 1s and 0s depend on the column they're in. We're used to this from the old htu (hundreds, tens and units) days of primary school: a 1 in the tens' column meant "one ten", whereas a one in the hundreds' column meant 100.

So:

h t u
1 0 0

was very different from:

h t u
1 0

It's much the same in binary. Every column has its value. Figure I shows these values for each column of a byte. Notice how the first column is column number 0 – micros start counting at 0.

To arrive at the value of a binary number, you just add together the values of the columns containing ones. Figure II shows how the %10101100 above translates into 172.

Once you get back the knack, binary numbers are quite easy to handle. We discuss some further techniques on Page 58.

One drawback to using binary from a human's point of view is that it's extemely easy to make mistakes with.

Column number	7	6	5	4	3	2	1	0
Value	128	64	32	16	8	4	2	1

Figure I: Binary column values

Value	128	64	32	16	8	4	2	1
Bit	1	0	1	0	1	1	0	0

= 128 + 32 + 8 + 4
= 172

Figure II: Translating binary

All those 1s and 0s can be very confusing, and you can end up writing:

%10110101

when you mean:

%10101101

It can be very difficult to spot where you've gone wrong.

When I deal with binary I always split the bits up into groups of four by putting a wiggly line down the middle. For example, I would write %10110101 as %1011ʃ0101

It's far easier for my eyes to see the pattern of the two sets of four than the whole eight at once. These sets of four are called nybbles. The left hand nybble, which contains the larger column values, is called the most significant nybble (MSN). The right hand one is called, not surprisingly, the least significant nybble (LSN).

Figure III shows what goes on with the least significant nybble.

Column	3	2	1	0
Value	8	4	2	1

Figure III: LSN values

I think that you'll immediately see that the biggest number it can hold is 15, when all the bits are 1. And, of course, the smallest number is 0. Figure IV shows all the patterns:

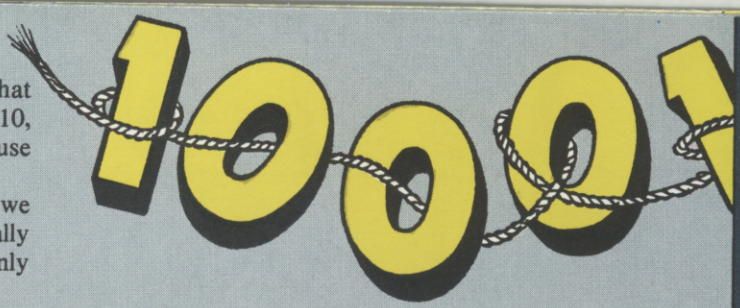
nybble pattern	value
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	10
1011	11
1100	12
1101	13
1110	14
1111	15

Figure IV: Nybble pattern plus values

You've probably thought to yourself, "So what? Why should I write %1111 when it's easier to write 15?" And you've got a point. In fact we're going to make it easier than that – we're going to give each of our nybble patterns a code, consisting of a single character.

The code for %0000 will be 0.
The code for %0001 will be 1.
The code for %0010 will be 2.

Yes, you've guessed it, the code is the value of the



pattern! So the code for %1001 will be 9. But what about when we come to %1010? This has the value 10, but we can't use that for the code, since we can only use one digit.

All right then, it's our code, so let's decree that we use A as the code for %1010. Okay, I know it's really ten and you know it's really ten, but our rule says only one character in the code so A it is.

You won't be surprised to learn that %1011, which has value 11, will have code B, %1100, which has value 12, has code C, and so on up to %1111 value 15, which has code F.

Figure V shows all the nybble patterns, this time with their codes and values.

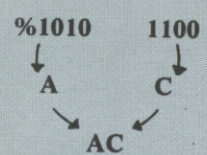
nybble pattern	code	value
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	A	10
1011	B	11
1100	C	12
1101	D	13
1110	E	14
1111	F	15

Figure V: Nybble patterns plus codes

What we've really done is to invent a rule that says, instead of counting 1, 2, 3 . . . 9, 10, 11, 12, 13, 14, 15 we count 1, 2, 3 . . . 9, A, B, C, D, E, F. For the moment, ignore what happens when we get to 16.

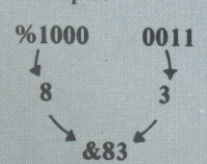
Of course the micro deals in bytes, but we can use our new code for these, too. After all, the most significant nybble is a pattern of four bytes, just like the least significant nybble, so all we do is code each nybble separately, but in the same manner.

Here's how:



If you look back to Figure II you'll see that %10101100 is 172, so AC is really another way of coding 172. To show that AC is really a number coded in our new way, we'd better prefix it with &, as we did with % for binary coding.

Here's another example:



Here %10000011 is 131, and coded in our new way this is &83. Notice how vital the "&" is. If we didn't have it we would mean 83 in our ordinary system of numbering – called denary or decimal – that is, 8 tens and 3 units and not 131 as we had intended.

So with this new method of coding you can code any byte in two characters, one for each nybble. For example 255 would be &FF. See if you can work out why.

This form of coding is called hexadecimal – hex for short. You'll understand why in a minute. Just as each column has a value in binary coding, so each has a value in hexadecimal. Let's see what they are.

The binary for 15 is:

%00001111 —> hexadecimal &0F

The binary for 16 is:

%00010000 —> hexadecimal &10

What's going on? Well, to obtain 16 from 15, we have to add one, so our hex sum is:

&0F +
&01
&10

What's happening is that we "put 0 down, carry one" when we get to 16. In normal sums we do this when we get to 10 – and the 1 goes in the tens column. In hexadecimal we carry when we get to 16 – each unit in the second column is worth 16. That's why I said don't worry about what happens when we get to 16. When we get there, we carry.

This, incidentally, is why it's called hexadecimal – the hex is for six, the decimal for ten – hexadecimal, the number code based on sixteen.

This means that:

&10 is 1 sixteen + 0 units = 16
&17 is 1 sixteen + 7 units = 23
&1B is 1 sixteen + 11 units = 27
&20 is 2 sixteens + 0 units = 32
&3C is 3 sixteens + 12 units = 60
&A3 is 10 sixteens + 3 units = 163

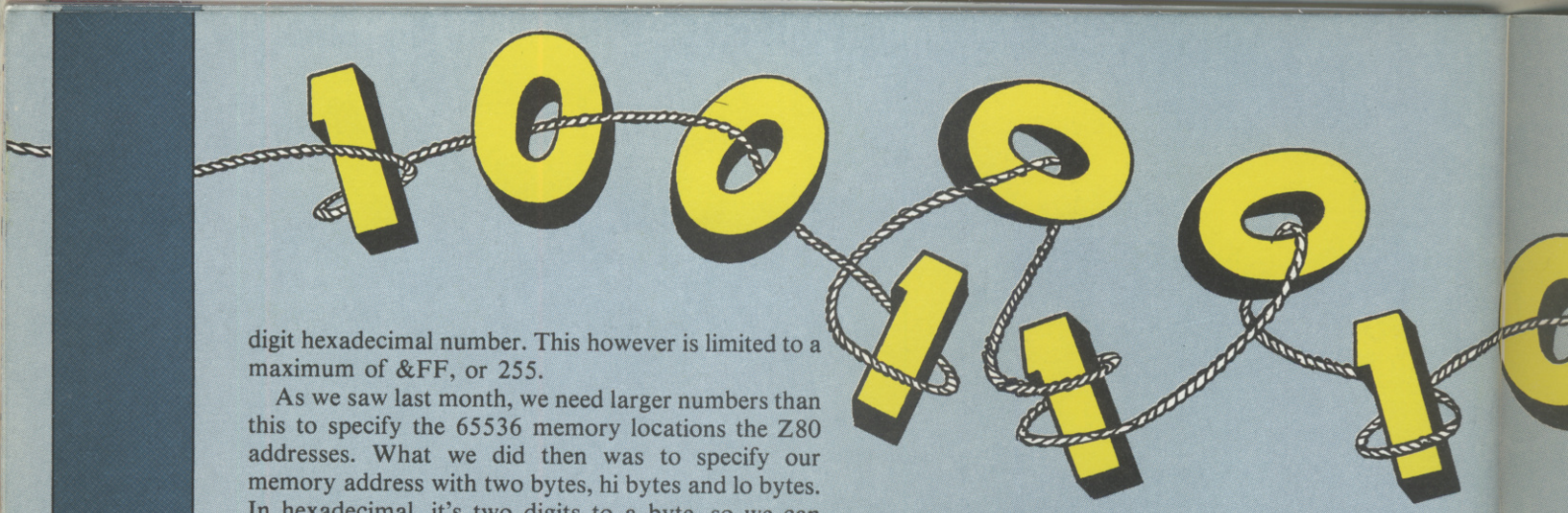
and so on . . .

That 16s column can present you with some hard sums, so Figure VI provides a sort of ready reckoner for you, telling you the value of a hex digit for each column.

Fine, we can now write the value of a byte as a two

Hex Digit	Value in 16's column	Value in 1's column
0	0	0
1	16	1
2	32	2
3	48	3
4	64	4
5	80	5
6	96	6
7	112	7
8	128	8
9	144	9
A	160	10
B	176	11
C	192	12
D	208	13
E	224	14
F	240	15

Figure VI: Hexadecimal Ready Reckoner



digit hexadecimal number. This however is limited to a maximum of &FF, or 255.

As we saw last month, we need larger numbers than this to specify the 65536 memory locations the Z80 addresses. What we did then was to specify our memory address with two bytes, hi bytes and lo bytes. In hexadecimal, it's two digits to a byte, so we can specify any address with four digits (instead of the cumbersome 16 bits we used last time).

Figure VII shows how it works:

Column	hi byte		lo byte	
Number	3	2	1	0
Value	4096	256	16	1

Figure VII: Four digit hex column values

Just as in binary we kept doubling our column values, so in hex we keep multiplying by 16. Hence the following values apply:

column 1	1 = 1
column 2	1 × 16 = 16
column 3	16 × 16 = 256
column 4	256 × 16 = 4096

This is quite sensible when you think about it. If the lo byte were &FF (255) – that is, as big as a single byte can hold – and we added one, we'd have to show somehow that we'd arrived at 256. We can do this by adding a one to the first column of the hi byte (which is worth 256) and setting the lo byte to &00 – in other words, put 0 down and carry 1.

The sums might get a bit hard if your mental arithmetic's a bit rusty, but you can translate from four digit hex addresses quite straightforwardly. For instance:

&0202 = 2×256 + 2	= 514
&1A10 = 1×4096 + 10×256 + 1×16	= 6672
&FFFF = 15×4096 + 15×256 + 15×16 + 15×1	= 65535

Anyway, enough of this number theory. Let's do some machine code. Last month we discovered that all machine code was was a sequence of bytes in memory that gave the micro certain tasks. All you did was point the micro at the first byte of the machine code and say go.

That gives us three problems. First of all we have to decide where in memory to put our machine code. Secondly we have to actually put the correct sequence of bytes into those memory locations. Lastly we have to tell the micro to go and do the machine code instructions!

To answer the first point, we're going to store the machine code at &7000 – 28672 in decimal. Never mind exactly why for the moment.

Secondly, to get the correct value bytes in memory we use the POKE command. The rather inelegantly termed POKE needs two parameters, or numbers to go with it, separated by a comma. It needs the location we want altering, followed by the number we want it altering to. So:

POKE 28672,201

will change memory location &7000 to &C9 (&C9 is the hex equivalent of 201). Try it. As nothing seems to have happened, the more sceptical of you might be wondering if I'm telling the truth. Has the value of the contents of memory location &7000 really been altered to &C9 by that POKE? Well, we can use a command called PEEK to examine the contents of memory locations. Try:

PRINT PEEK 28672

You'll get the answer 201. So the POKE really has worked. Now, believe it or not, we've just entered a one byte machine code program into memory. You see, &C9 is the instruction, or opcode, that tells the Z80 to "return from whence you came". Its mnemonic – an easily remembered "codeword" – is RET. We met it on on Page 34 in Part I.

Right, we've put our machine code program at &7000. We get our Spectrum to execute it with:

USR 28672

This is a function which returns a value – which we'll ignore for the moment. It's a sort of "GOSUB from Basic to the machine code routine that starts at 28672". That's why we need the RET (&C9) there. It's the "RETURN" to match the "GOSUB" of the USR command. Try it now. Enter:

LET X=USR 28672

The micro "gosubs" to &7000 (28672) and does what the byte tells it. In this case it merely says "go back", and you end up back in Basic. All right, nothing spectacular has happened – but it worked, so congratulations. You've just run your first machine code program.

Let's try something that does a bit more than this though, to prove we're really running code. As we said last month, the firmware is full of machine code routines that keep the micro going. And those routines are there for you to call on – they're already written.

Let's use one. There's a routine that starts at &0D6B (3435 decimal) that clears the screen. To demonstrate it, let's call it directly from Basic by entering:

LET X=USR 3435

As you'll see, it works. What I want to do, though, is to

Memory address	28672	28673	28674	28675
Contents of memory location	205	107	13	201
Meaning of above bytes	CALL the subroutine at address specified by next two bytes	lo byte	hi byte	RETurn from whence you came
		Address of subroutine required by previous byte (lo byte first)		

Figure VIII: Our simple clear screen program

call this routine from one of my own machine code programs. After all, I might want to clear the screen in the middle of one of my magnificent machine code creations, and I don't want to have to go into Basic to do it.

The Z80 has a useful instruction that lets us call (or "gosub") to another machine code routine – the code for it is &CD (205). You give the Z80 the address you want it to "gosub" to in the two bytes directly following the &CD in memory. So to clear the screen with the routine at &0D6B you might think we want the following sequence of bytes in memory:

CD 0D 6B
(205 13 107)

You'd be wrong though. The Z80 expects memory addresses to be given to it "back to front" – that is, lo byte first, followed by hi byte. So to call the routine at &0D6B the sequence is:

CD 6B 0D
(205 107 13)

The mnemonic for &CD is CALL, by the way. However, we've missed out something very important in our program – the RET (&C9). So our complete sequence of bytes, starting at memory location &7000 (28672), would be:

CD 6B 0D C9
(205 107 13 201)

Figure VIII illustrates the program.

Now let's get the program into memory. Enter:

```
POKE 28672,205
POKE 28673,107
POKE 28674,13
POKE 28675,201
```

If you like, you can check up on it with PEEK to see if it's there. When you are ready, call your routine with:

LET X=USR 28672

and your screen should clear. Well done – your first machine code program that actually has a visible effect. Type some rubbish on to the screen, then:

LET X=USR 28762

again. As you'll see, your routine is still there. Now take care with your machine code programs – it's all too easy to crash your machine or wipe out your program. For instance, try:

LET X=USR 0

and you'll see what I mean. If you now try the routine, you'll see that your code has gone. It's often tiny errors that cause such disasters. The solution is simple – check and recheck before each Enter.

As I mentioned last time, you can get special programs called assemblers that let you type in your programs in the more meaningful mnemonics rather than as a list of bytes. If you wrote the above program on an assembler and then listed it, it would look something like Figure IX.

address	opcodes	mnemonics
7000	CD 6B 0D	CALL &0D6B
7003	C9	RET

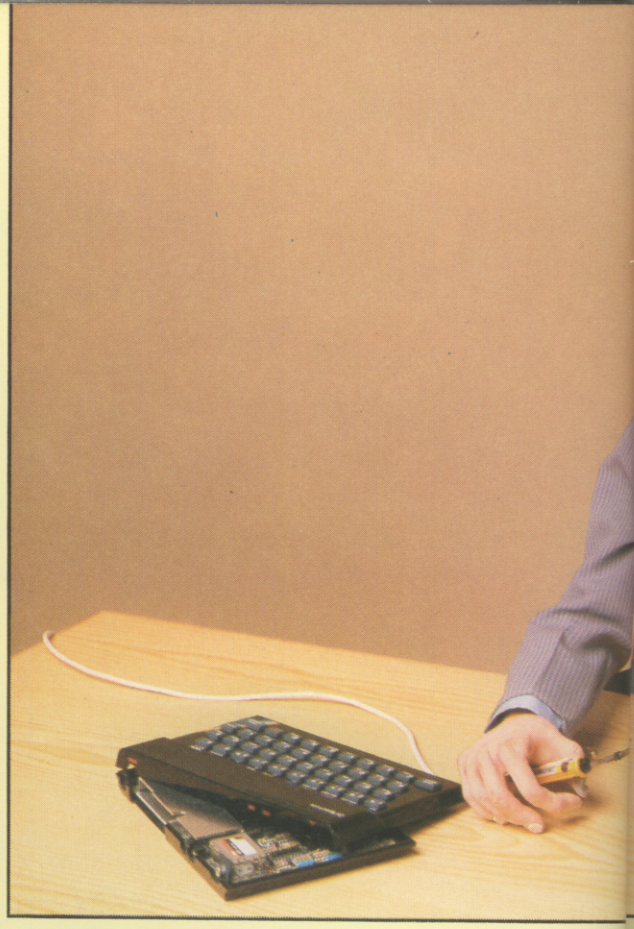
Figure IX: Assembler listing

The mnemonics are what would be typed into the assembler. The address column shows the address of the starting byte for each instruction, and the opcodes column shows the opcode and associated data bytes for each instruction.

● ***Well that's all for this time. In the next part of the series we'll be looking at better ways of entering machine code – and learning some new instructions.***

RAM upgrade

Stuck with 16k? We help point the way to adding extra memory to earlier Spectrums



THE 16k version of the Spectrum has not been manufactured for some time, but there are still some about. If you have one, you have no doubt noticed that there are very few commercially available programs that fit in the smaller memory, and that the chart-topping games all need 48k.

Luckily there are plenty of suppliers of extra memory for the Spectrum, and you can upgrade your junior model to the full 48k for little over £20.

DIY or send away?

DIY kits are the least expensive way to upgrade, but require a measure of manual dexterity to fit. Mostly you only have to plug chips into sockets. Not quite as easy as it sounds, but fairly easy – but you may have to solder a few links to the Spectrum circuit board. A really sloppy job here could cause trouble, but it is a task that should not be beyond anyone.

Opening the computer case is a simple matter of removing five screws from underneath. Unfortunately this act will void your Sinclair warranty, but this has probably already lapsed in the case of a 16k model.

Some suppliers of DIY memory upgrades – and some Spectrum repair specialists – will fit the chips for you for a bit extra. In this case, you should get a new warranty thrown in. Don't expect the man at the computer counter in your local chainstore to fit them while you wait, however. You'll have to send your computer away.

Self fitting

HAVING removed the screws from the case you will find that the top of the case with the keyboard is attached only by the keyboard ribbon cables. These flat, printed film cables should be pulled gently out of their sockets and the top of the case set aside. The cables are rather fragile and must be handled with care, particularly during reassembly.

There are two main things to watch when fitting chips into sockets. The first of these is orientation. There are two ways in which the chip can be inserted, the right way round and the wrong way round. Even people with a PhD get them in the wrong way round on occasion, so read the instructions and pay attention!

The chips will be clearly marked with a dot or notch at one end, and the instruction sheet will make it very plain which end goes where. Not all of the chips in the kit will be the same, so it is also possible to put them in the wrong sockets – but even a PhD wouldn't be that careless.

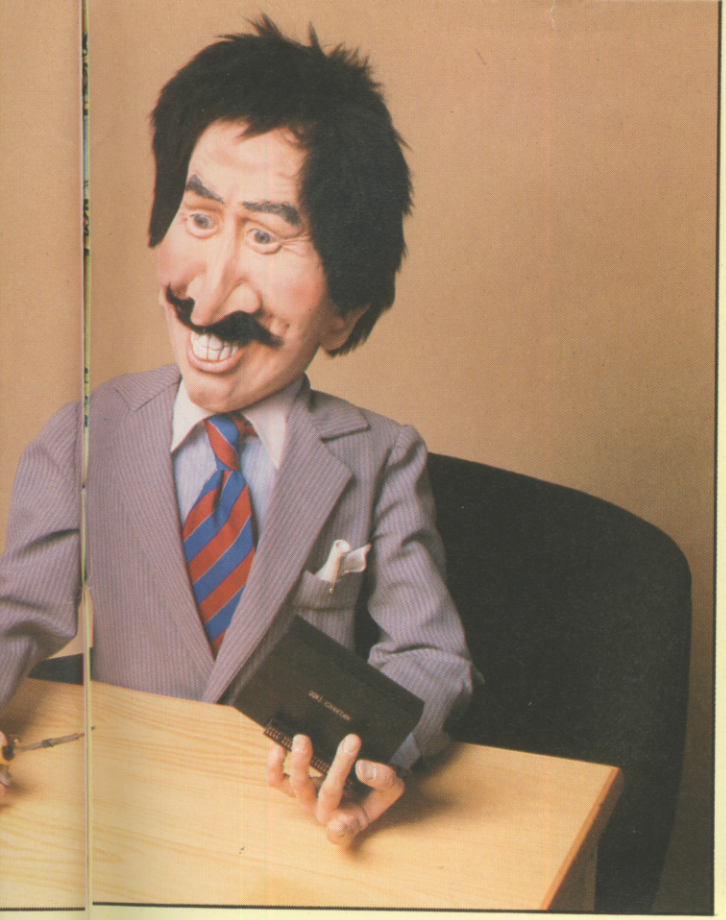
The second problem with fitting chips is that they always seem to be supplied with slightly splayed legs. This makes it difficult to line up the two rows of legs with the corresponding holes in the sockets. Carelessness or a heavy hand at this point can lead to bent or broken legs, and a crippled chip. An IC insertion tool is a perfect solution to this difficulty if you can borrow one from somebody, but they are too expensive to be worth buying for just this one job.

The alternative is to bend the legs into line by holding the body of the chip and pressing each row of legs in turn against a hard, flat surface. This is a tricky procedure, as too much bending is as bad or worse than too little, but if you take it slowly and in stages, offering the chip up to the socket to check for fit, it will be easy enough. By the time all the chips are in place you will feel that insertion tools are toys for the inexperienced.

You may or may not need to solder a link on the computer's circuit board. This will depend on the chips used in the upgrade kit. In the early Spectrum, Sinclair found that he could save a lot of money by buying partially faulty memory chips, and wiring them up to use just the good half of the chip. The link was necessary to select which half of the set of chips was to be used.

Some of the memory upgrades continue this practice and therefore need the link, but the price of fully functional chips has dropped to the point where there is no longer a big saving, so some upgrades use fully functioning chips and do not need the link.

If you don't want to solder, make sure that the kit you buy promises "no soldering". If you do intend to solder, then you will need a suitable iron, which means not too big. About 15 to 25 watts will do. You will also need proper resin cored electrical solder, and a suitable length of tinned wire without insulation – make sure it



isn't enamelled – for the link.

If you have never soldered before, then what you will need most is practice. Try out the tools on some waste materials until you get the feel of it, and remember – it's easier to get the solder on to things than to get it off again!

You may have heard horror stories about chips being rendered defunct by static discharges arising from simply handling them. These stories are true! However they apply mostly to the CMOS type of chips which are not used in the Spectrum. Elaborate anti-static procedures are therefore not necessary, but it does no harm to avoid nylon clothing.

The final and most important step is to double check everything visually before switching on. A chip may survive being powered up in the wrong socket but if it is the wrong way around, or has a leg bent under, it could just expire. Check first.

A notable exception

If you have the very early Issue 1 Spectrum, then it is not at all easy to fit an upgrade internally, so don't try. You would just destroy its antique value. Issue 1 Spectrums can be identified by the fact that the rubber keys are grey rather than bluish. If you still aren't sure, open it up and you will find the issue number printed on the circuit board.

You will also notice that it isn't fitted with a flock of empty sockets, but only two, which is why it is difficult to fit the chips. In fact these sockets are meant for fitting an upgrade circuit board that used to be supplied by Sinclair. Issue 1 hasn't been made since 1982 though, so there aren't that many about.

The Rampak solution

IF you don't fancy opening up your computer or sending it away (or if you have the rare Issue 1), there is a memory upgrade from Cheetah that just plugs on to the edge connector like a joystick interface. This costs quite a bit more than buying the bare chips and installing them yourself – almost double in fact – but it is certainly simple. Naturally, the edge connector is carried through.

Whatever you decide, do not attempt to use an old Rampak left over from a ZX81. It not only won't work, but will cause damage.

Some suppliers

Cheetah Rampak
24 Ray Street
London EC 1R 3DJ
01-833 4909

Micro-World Computers
Repairs and Memory Upgrading
1006/1010 Manchester Road
Linthwaite
Huddersfield HD7 5QQ
(0484) 846117

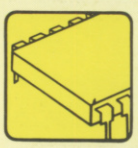
Mancomp
Repairs and memory upgrading
Printworks Lane
Levenshulme
Manchester M19 3JP
061-224 1888/061-224 9888

Video Vault
Repairs and memory upgrading
140 High Street West
Glossop
Derbyshire
(04574) 66555/67761

ELR International
Upgrade kit, no soldering
St. Nicholas House
The Mount
Guildford
Surrey GU2 5ZHN
(0483) 505605

RAM Electronics
Upgrade kit
From branches of Boots, Menzies, Spectrum Group, or
Unit 8
Redfields Industrial Park
Redfield Lane
Church Crookham
Aldershot Hants GU13 0RE
(0252) 850085

Fox Electronics
Upgrade kit
Fox House
35 Martham Road
Hemsby
Great Yarmouth
Norfolk NR29 4NQ
(0493) 732420



Looking into books

IOLO DAVIDSON looks along his shelf and picks out six of the best

THERE are a great many books available that claim to introduce the beginner to the Spectrum in an easy, painless fashion. Some of them are useful but many are a waste of time, offering little that you could not find in the manual that came with the computer. At worst they contain only listings of moronic games written in Basic, riddled with bugs and printing errors. Good or bad, computer books always seem to cost the earth, too.

The "Tune a Day" approach is not well suited to computing. Computing is complicated. It is one thing to attempt to illuminate a complex subject, and quite another to pretend that the subject can be made simple. Many people have learned to program by struggling with badly printed magazine listings, whereas others have learned in a more formal way in the classroom, but in both cases the real teacher is the computer itself. For a book to aid this process it must contain real, hard information, not just optimism and encouragement. A good computer book is primarily a work of reference.

Rather than recommend a list of worthy books, I give details below of the books I actually use myself. They are not necessarily aimed at the beginner, and are all the better for that. After all, no one wants to stay a beginner.

Sinclair ZX Spectrum BASIC Programming
(The Original Spectrum Manual)
By Stephen Vickers, Editor Robin Bradbeer
Publisher Sinclair Research

The User Guide that comes with the Spectrum Plus is very beautiful, but does not cover the subject anything like as well as the manual that came with the original rubber-key Spectrum. This one has 232 pages, none of them in colour, covering the Spectrum's Basic in tutorial detail, plus important aspects of using machine code, such as the in/out ports and system variables. The best beginner's guide and an indispensible reference.

Programming The Z80
By Rodney Zaks
Publisher Sybex

This one is something of an expensive luxury, unless you really need to know the number of T-states consumed by a machine code instruction. There is nothing here about the Spectrum, but absolutely everything about the Z80 microprocessor.

The Complete Spectrum ROM Disassembly
By Dr Ian Logan and Dr Frank O'Hara
Publisher Melbourne House

Just as the title says, this is the contents of the Spectrum ROM, the machine code program that runs the Spectrum, presented as a Z80 assembly language listing with extensive annotations explaining what is going on throughout. You won't need this one, or understand it, unless you are already into machine code programming. Then you won't be able to live without it.

Understanding Your Spectrum
By Dr Ian Logan
Publisher Melbourne House

This book is an excellent way of making the transition from Basic to machine code, and afterwards it will serve as a vital reference for machine code programming. The section on calling machine code routines in the Spectrum ROM from your own programs is especially useful. After years of hard use the cover is coming off my copy. I must get another.

Spectrum Hardware Manual
By Adrian Dickens
Publisher Melbourne House

A soldering iron wielder's guide to the Spectrum, covering fault finding and fixing, experiments, and add-on projects. Definitely not for the faint of heart, but you can't blow your computer up just by reading about it. Perhaps most valuable are the circuit diagrams of various sections of the computer, such as the power supply, and one of the whole Spectrum. My early copy only covers Issue 1 and Issue 2 Spectrums, so make sure you get a suitable edition if yours is an Issue 3 or later.

Master Your ZX Microdrive
By Andrew Pennell
Publisher Sunshine

Andrew Pennell actually uses Sinclair's microdrives himself, and does not gloss over the problems and bugs. There are lots of example programs, in both Basic and assembler, and a good section on using machine code with the microdrives. Not a must by any means, but the definitive work on this subject.

The COMPLETE

SPECTRUM

With Part II, you will see how The Complete Spectrum is developing into a richly-detailed encyclopaedia covering all aspects of computing with your Spectrum.

To keep your copies in perfect condition we have produced an attractive binder to hold all six parts. It is bound in smart black pvc, with the logo printed in silver. All you have to do before inserting the monthly parts in your binder is to remove the outer cover and the 'Special Offers' section in the centre.

For addresses in the UK, the binder costs £3.95, or £6.95 for overseas.

If you missed Part I, or would like to subscribe for all the remaining parts, you can do so by using the order form in the centre of this issue.

Contents of Part I included:

● A step by step guide to your first 30 minutes using the Spectrum. ● Advice on saving and loading. ● How to avoid errors when typing in listings. ● An investigation of the Z80, the chip behind it all. ● Computer jargon clarified. ● And the start of our regular series – a simple guide to basic Basic, an introduction to machine code, and a how-to-do-it guide to graphics.

In Part Three...

We help you explore further into the ever-growing world of the Spectrum!

Databases – what they are and how to get the best from them.

Printers and interfaces – why you need them, and how they work.

Speech – how it can give an extra dimension to your Spectrum.

Upgrades – we compare all the latest Spectrum keyboards.

Breakdowns – what to do if your Spectrum ever lets you down.

And there are the regular features on Basic, machine code, and colourful articles on graphics and animation techniques.

Plus pages of hints and tips to help you get the best from your Spectrum.

On sale March 7

RAMBO™

FIRST BLOOD PART II™

OFFICIAL COMPUTER GAME OF THE BOX-OFFICE No.1 SMASH HIT. TAKE THE ROLE OF SYLVESTER STALLONE IN ALL THE ACTION-PACKED BATTLE SCENES.

RELEASED NOW

STALLONE

©1985 Anabasis Investments N.V. All rights reserved
TM a Trademark of Anabasis Investments N.V. Licensed by Stephen J. Cannell Productions

SPECTRUM 48k **7·95**

ocean

COMMODORE 64 & AMSTRAD **8·95**

Ocean House · 6 Central Street · Manchester · M2 5NS · Telephone 061 832 6633 · Telex 669977

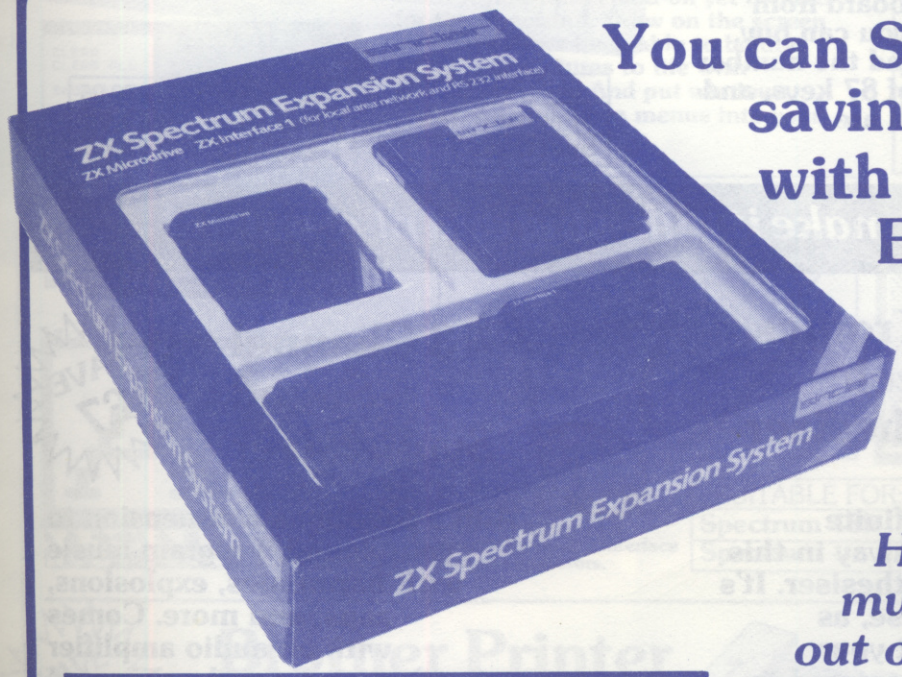
Ocean Software is available from selected branches of: Boots, WHSMITH, John Menzies, WOOLWORTH, LASKYS, Rumbelows, Greens, Spectrum Shops and all good software dealers. Trade enquiries welcome.

EXCLUSIVE!

SAVE

£10 £5 £5 £50 £10 £10 £10 £10 £10

on best-selling Spectrum products with these unbeatable bargains!


ZX Spectrum Expansion System
ZX Microdrive ZX Interface 1 (for local area network and RS 232 interface)
ZX Spectrum Expansion System

You can SPEED UP saving/loading with Sinclair's Expansion System

Rec. price	£99.95
Our price	£69.95
SAVING	£30.00

Helps you get much-more fun out of computing on your Spectrum!

LOOK WHAT YOU GET!

- One ZX Microdrive. ● One ZX Interface 1.
- Fun-packed Microdrive demonstration cartridge.
- Three blank cartridges.
- Connecting lead for Microdrive/Interface 1.
- ZX Net lead. ● Full documentation.

SUITABLE FOR:	
Spectrum 48k	✓
Spectrum +	✓

All prices in these pages include VAT + post and packing

Please order using the coupon at the back

Table I: INK colours

INK (foreground) colour to determine the new background colour.

Enhance your Spectrum with a new keyboard!

SAVE £10

Top-selling DK'Tronics keyboard

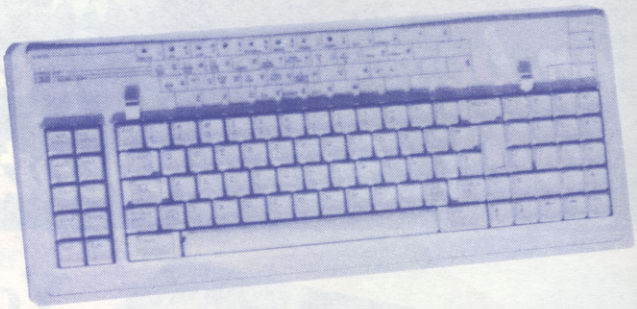
For ease of use nothing can beat this Microdrive-compatible keyboard. It offers more keys and functions than any other in its price range – and puts your humble Spectrum into the professional league! Fitting it only takes a few minutes, a screwdriver and the simple instructions supplied.

SUITABLE FOR:	
Spectrum 48k	✓
Spectrum +	✓

Rec. price	*£37.50*
Our price	£27.50
SAVING	*£10.00*

SAVE £10

The Saga 3 Elite Keyboard



Give your Spectrum the sheer elegance it deserves with this top-of-the-range keyboard from Saga. It's the finest you can buy. With the keyboard and the number pad you get a total of 87 keys, and a massive 27 of them are auto-shifted.

SUITABLE FOR:	
Spectrum 48k	✓
Spectrum +	✓

Rec. price	*£79.95*
Our price	£69.95
SAVING	*£10.00*

. . . and make it talk, sing and play music!

SAVE £5

DK'Tronics Speech Synthesiser

There's an almost infinite vocabulary tucked away in this popular speech synthesiser. It's extremely easy to use, as everything you want your Spectrum to say is entered in normal English. The result is realistic speech that is instantly recognisable.

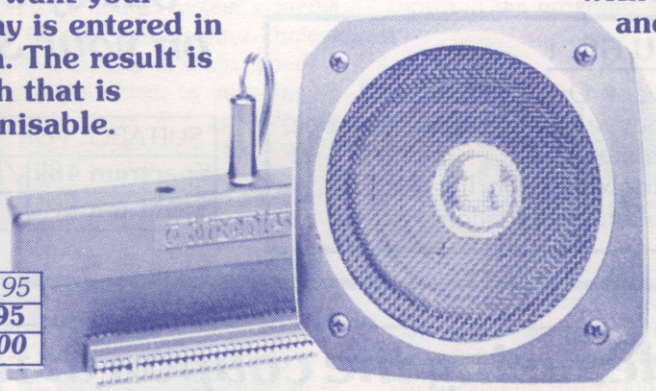
SUITABLE FOR:	
Spectrum	✓
Spectrum +	✓

Rec. price	*£24.95*
Our price	£19.95
SAVING	*£5.00*

3 Channel Sound Synthesiser

SAVE £7

Add a totally new dimension to your sound. Program music with harmonies, explosions, zaps, and more. Comes with an audio amplifier and a pad – mounted 4" speaker.



SUITABLE FOR:	
Spectrum	✓
Spectrum +	✓

Rec. price	*£29.95*
Our price	£22.95
SAVING	*£7.00*

2

way it must be followed by a semi-colon, otherwise the Spectrum gets confused and generates an error.

The second command associated with colour is

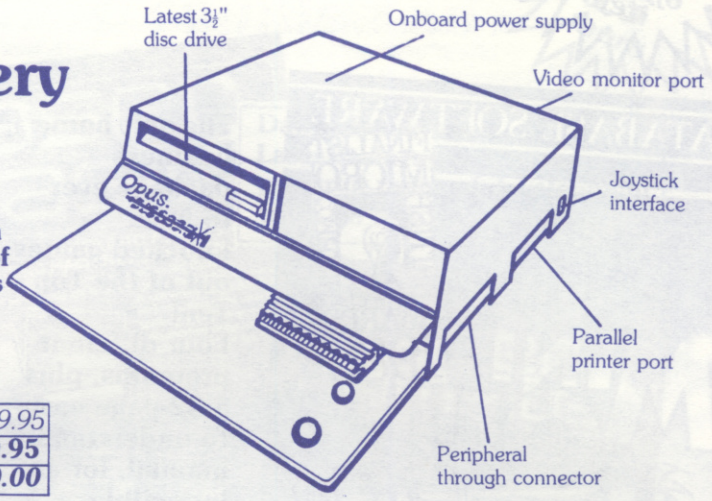
```
20 PAPER INT (RND*8)
30 PRINT "This is a message"
40 GO TO 10
```

More savings on drives, printers, mice!

SAVE £50

Opus Discovery

In one simple step you can transform your Spectrum into a complete computer system. It combines a powerful disc drive with a wide range of interfaces capable of running printers, monitors, joysticks and other peripherals. No other system can offer your Spectrum so many options.



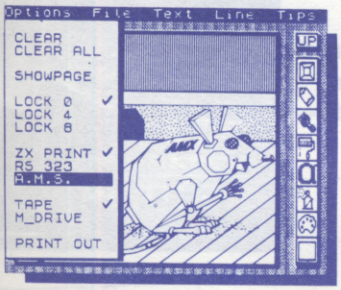
SUITABLE FOR:	
Spectrum 48k	✓
Spectrum +	✓

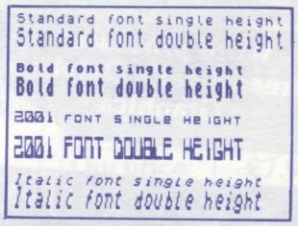
Rec. price	£199.95
Our price	£149.95
SAVING	£50.00

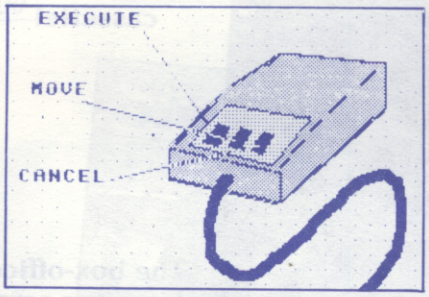
SAVE £10

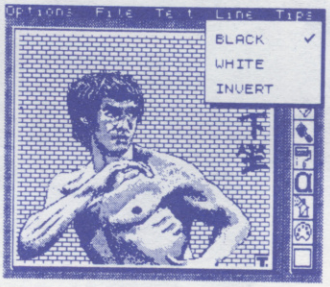
AMX MOUSE

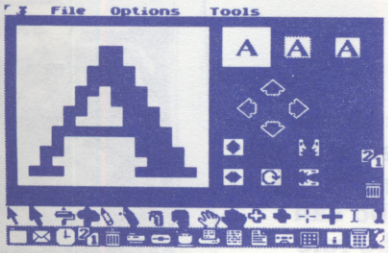
Put yourself in the picture with the most revolutionary add-on yet created for the Spectrum. Draw on the screen as you've never been able to before. Bring your pictures to life with glowing colours. And put windows, icons and pull-down menus into your own programs!




Standard font single height
Standard font double height
Bold font single height
Bold font double height
2001 FONT SINGLE HEIGHT
2001 FONT DOUBLE HEIGHT
Italic font single height
Italic font double height







Package comes complete with AMX Mouse, graphics software, mouse port, centronics printer interface and all connectors.

SUITABLE FOR:	
Spectrum 48k	✓
Spectrum +	✓

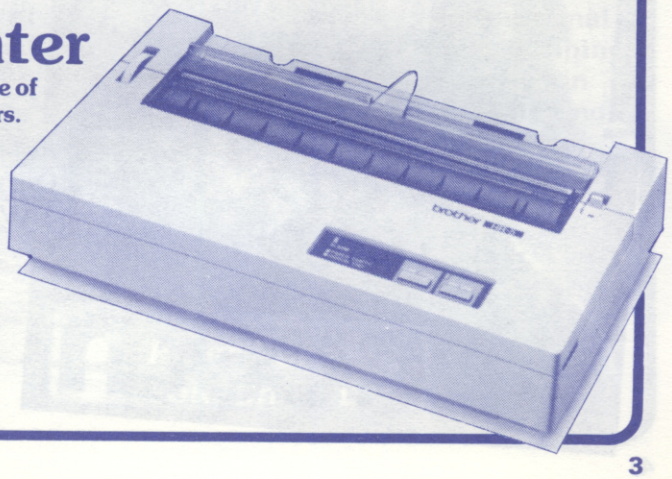
Rec. price	£69.95
Our price	£59.95
SAVING	£10.00

SAVE £10

Brother Printer

A feature-packed printer from one of Japan's best-known manufacturers. It prints on both thermal and ordinary plain paper.

● Speed 130cps. ● Up to 132 characters per line. ● Bi-directional. ● Takes A4 size paper. ● Mains or battery operated. ● 4 fonts with reduced, enlarged and bold modes. ● Requires centronics interface – see offer overleaf.



SUITABLE FOR:	
Spectrum	✓
Spectrum +	✓

Rec. price	£99.95
Our price	£89.95
SAVING	£10.00

3

Table I: INK colours

INK (foreground) colour to determine the new background colour.

SAVE or £2 more off each item

More cut-price s

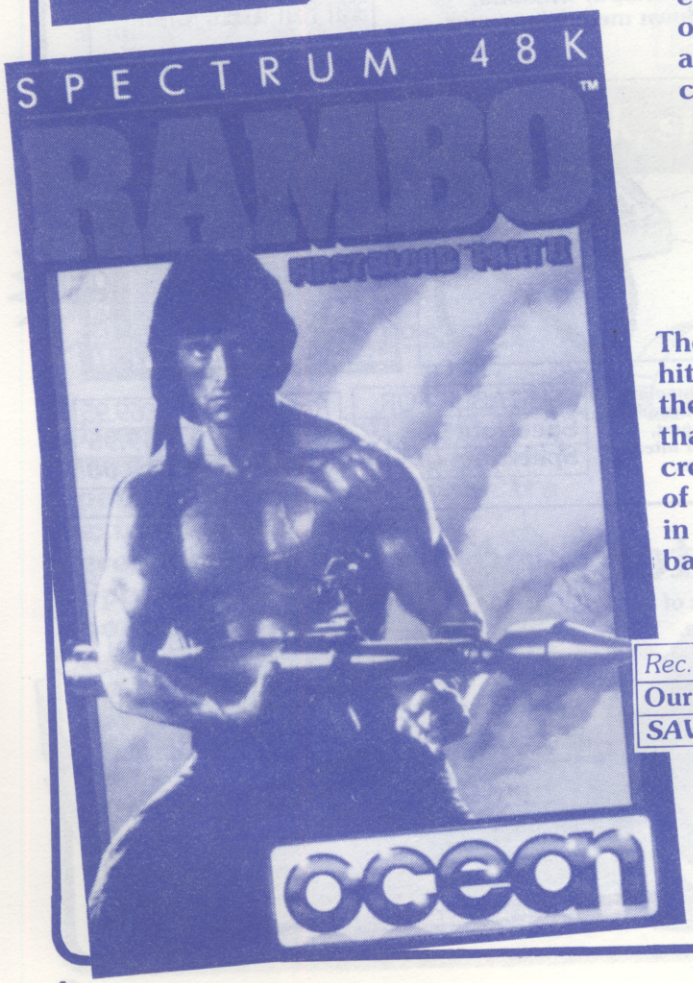
The only home business package ever to have knocked games out of the Top Ten!
Four different programs, plus a 32-page easy-to-understand manual, for an incredible price.

Rec. price	*£5.95*
Our price	**£3.95**
SAVING	***£2.00***

Word processor
Spreadsheet · Database
Graphics

Spectrum 48K
Spectrum

Four classic games every Spectrum owner will want – and all on one cassette!



The box-office smash hits your screen with all the high-energy drama that only Rambo can create. Take the role of Sylvester Stallone in explosive all-action battle scenes!

Rec. price	*£7.95*
Our price	**£5.95**
SAVING	***£2.00***

4

way it must be followed by a semi-colon, otherwise the Spectrum gets confused and generates an error.

The second command associated with colour is

```
20 PAPER INT (RND*8)
30 PRINT "This is a message"
40 GO TO 10
```

e software offers

Order any 5 and SAVE £5 more

Rec. price	£9.95
Our price	£7.95
SAVING	£2.00

Dawn of a new age!
Laser Basic is the first in a powerful range of development tools for fast programming.

Rec. price	£14.95
Our price	£10.95
SAVING	£4.00

Create your own machine code programs!
So easy – just 'load and go!'

Rec. price	£9.95
Our price	£7.95
SAVING	£2.00

The chart-topping 3 part adventure comes on two cassettes – and will give you many hours of enjoyment!

Rec. price	£9.95
Our price	£7.95
SAVING	£2.00

All the skill and tension of international show jumping captured in this gripping simulation.

Rec. price	£7.95
Our price	£5.95
SAVING	£2.00

Please order using the coupon at the back

5

Table I: INK colours

INK (foreground) colour to determine the new background colour.

Save on joysticks, light pens and interfaces

Kempston Formula 1 Joystick and Interface

Undoubtedly the best joystick currently available. This latest version has eight-way micro switches for faster action and longer life. It's a firm favourite with games players because it helps achieve those high scores which before seemed just out of your reach!

Joystick only	
Rec. price	£16.95
Our price	£13.95
SAVING	£3.00

SUITABLE FOR:	
Spectrum	✓
Spectrum +	✓

Joystick and Interface	
Rec. price	£26.90
Our price	£18.90
SAVING	£8.00

DK'Troniks Light Pen

This best selling light pen plugs neatly into the rear of the Spectrum or the printer connector. The operating software now runs faster and supports 18 pre-defined instructions. ● Save and load partially complete screens. ● Accuracy down to pixel level. ● Change all colours (border, paper, ink). ● Draw circles, lines, boxes, arcs. ● Retain and animate screens. ● Insert text onto screen.

SUITABLE FOR:	
Spectrum	✓
Spectrum +	✓

Rec. price	£19.95
Our price	£14.95
SAVING	£5.00

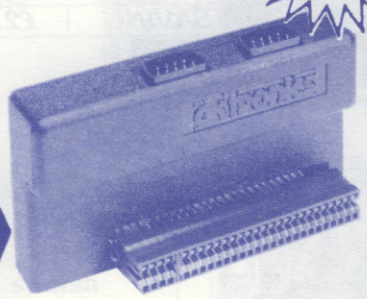
Programmable Interface

A fully programmable joystick interface that allows you to run ANY software from ANY supplier. Easy programmed in seconds to give 8 directional joystick movements.

SUITABLE FOR:	
Spectrum	✓
Spectrum +	✓

Rec. price	£22.95
Our price	£18.95
SAVING	£4.00

Dual Port

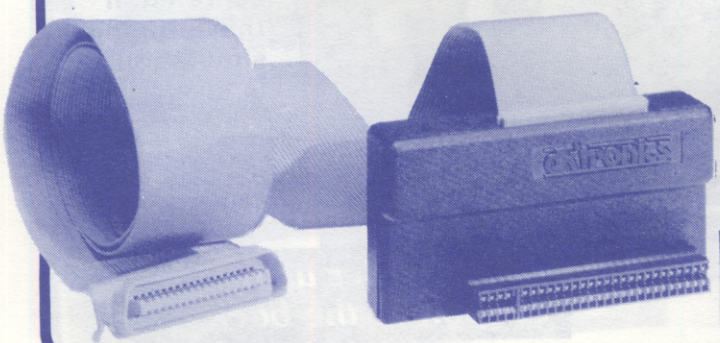


You can use two joysticks simultaneously with this Dual Port interface. It will run ANY software and accept ANY Atari-type joystick.

SUITABLE FOR:	
Spectrum	✓
Spectrum +	✓

Rec. price	£13.00
Our price	£10.00
SAVING	£3.00

DK'Tronics Parallel Centronics Interface



Enables you to link your Spectrum to any printer with a standard centronics output, such as the Brother HR5 offered overleaf. It is fully Microdrive compatible and supplied complete with full instructions.

SUITABLE FOR:	
Spectrum	✓
Spectrum +	✓

Rec. price	£39.95
Our price	£29.95
SAVING	£10.00

6

way it must be followed by a semi-colon, otherwise the Spectrum gets confused and generates an error.

The second command associated with colour is

```
20 PAPER INT (RND*8)
30 PRINT "This is a message"
40 GO TO 10
```

A dust cover for your Spectrum and Spectrum+

Made of soft pliable water-resistant black vinyl and decorated with *The Complete Spectrum* logo in silver.

ONLY £3.95

A binder for The Complete Spectrum

Keep all your copies neat and clean in this handsome binder.

ONLY £3.95

Make sure you've got a COMPLETE set of

Use the form below to order the next four parts – delivered to you post free every month for just £6.

If you missed issue 1, we'll send it to you for £1.75.

Or you can order issue 1, together with issues 3 to 6, for only £6.95.

Valid to March 31, 1986

ORDER FORM

Please enter quantity as required in box

Please allow up to 28 days for delivery.

The Complete Spectrum

			£	p	
Parts 3-6:					
UK, Eire	£6.00	9013			*
Europe	£10.00	9014			*
Overseas	£18.00	9015			*
Issue No. 1:					
UK, Eire	£1.75	9016			
Europe	£2.25	9016			
Overseas	£2.75	9016			
*** Add an extra 95p and we'll also send you Issue 1.**		9020			

Binder

			£	p
UK	£3.95	9030		
Europe/Overseas	£6.95	9031		

Dust Covers

			£	p
Spectrum Dust Cover				
UK	£3.95	9035		
Europe/Overseas	£6.95	9036		
Spectrum+ Dust Cover				
UK	£3.95	9037		
Europe/Overseas	£6.95	9038		

Keyboards

			£	p
Saga 3 Elite (UK only)	£69.95	9041		
DK'Tronics Keyboard (UK only)	£27.95	9042		

ZX Spectrum Expansion Pack

			£	p
(UK only)	£69.95	9040		

Order at any time of the day or night

Telephone Orders: **061-429 7931**

Orders by Prestel: **Key *89, then 614568383**

Don't forget to give your name, address and credit card number

ENQUIRIES ONLY: **061-480 0171**

Peripherals

			£	p
Speech Synthesiser	£19.95	9043		
Dual Joystick port	£10.00	9044		
Programmable Joystick Interface	£18.95	9045		
3 Channel Sound Synthesiser	£22.95	9046		
Lightpen	£14.95	9047		
Parallel Centronics Interface	£29.95	9048		
Formula 1 Joystick	£13.95	9049		
Formula I Joystick and Interface	£18.90	9062		

Softwaret

			£	p
Laser Compiler	£7.95	9057		
Laser Basic	£10.95	9055		
Never Ending story	£7.95	9058		
They Sold a Million	£7.95	9059		
Rambo	£5.95	9060		
Show Jumping	£5.95	9061		
Mini Office	£3.95	9001		
†5 or more items – deduct	**£5.00**	9099		

Disc Drive

			£	p
OPUS DISCOVERY 1 Includes all interfaces and power supply (UK only)	£149.95	9100		

Printer

			£	p
BROTHER HR5 Includes 4 fonts and paper (UK only)	£89.95	9101		

AMX Mouse

			£	p
Complete with interface and software (UK only)	£59.95	9102		

All prices include postage, packing and VAT
Overseas orders sent exclusively by Air Mail

TOTAL

NOW PLEASE COMPLETE PANELS OVERLEAF

7

INK (foreground) colour to determine the new background colour.

Table I: INK colours

Payment: Please indicate method (✓)

☐ *Access/Mastercharge/Eurocard*

☐ *Barclaycard/Visa*

☐ *Cheque/PO made payable to Database Publications Ltd.*

BLOCK CAPITALS PLEASE

Name _____

Address _____

_____

_____

_____ Post Code _____

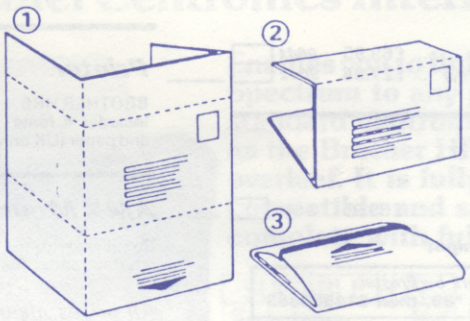
Signed _____

TUCK INTO FLAP

Don't forget to provide: Your address, your cheque/credit card No.

The Complete Spectrum
FREEPOST,
Europa House,
68 Chester Road,
Hazel Grove,
Stockport SK7 5NY.

NO POSTAGE NEEDED (IF POSTED IN UK)



way it must be followed by a semi-colon, otherwise the Spectrum gets confused and generates an error.

The second command associated with colour is

```
20 PAPER INT (RND*8)
30 PRINT "This is a message"
40 GO TO 10
```